โปรแกรมออกแบบระบบติดตั้งหัวล่อฟ้า บนเว็บแอปพลิเคชั่น

LIGHTNING PROTECTION DESIGN PROGRAM ON WEB APPLICATION

นายศักดิ์สิทธิ์ มหาราช รหัสนักศึกษา 46015374

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

โปรแกรมออกแบบระบบติดตั้งหัวล่อฟ้า บนเว็บแอปพลิเคชั่น

LIGHTNING PROTECTION DESIGN PROGRAM ON WEB APPLICATION

โดย

นายศักดิ์สิทธิ์ มหาราต รหัสนักศึกษา 46015374

อาจารย์ที่ปรึกษา

อาจารย์ อำนาจ ขาวเน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

ปริญญานิพนธ์ปีการศึกษา 2549

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง โปรแกรมออกแบบระบบติดตั้งหัวล่อฟ้า บนเว็บแอปพลิเคชั่น

Lightning Protection Design Program On Web Application

ผู้จัดทำ

1. นายศักดิ์สิทธิ์ มหาราช เลขประจำตัว 46015374

อาจารย์ที่ปรึกษา

(อาจารย์อำนาจ ขาวเน)

โปรแกรมออกแบบระบบติดตั้งหัวล่อฟ้า บนเว็บแอพพลิเคชั่น

นาย ศักดิ์สิทธิ์ มหาราต 46015374
อาจารย์ อำนาจ ขาวเน อาจารย์ที่ปรึกษา
ปีการศึกษา 2549

บทคัดย่อ

ในปัจจุบันเนื่องจากประเทศไทยมีการเจริญเติบโตทางด้านเทคโนโลยี เศรษฐกิจและอุตสาหกรรมมากขึ้น และพนักงานบุคลากรในแต่ละหน่วยงานต่างๆ ก็ต้องการความปลอดภัยในการทำงานในสถานที่ต่างๆ ซึ่งระบบรักษาความปลอดภัยของโรงงานอาคารนั้น ในส่วนที่เป็นด้านระบบคอมพิวเตอร์ หรือระบบควบคุมต่างๆและตัวพนักงานเอง ต้องการความปลอดภัยเป็นอย่างมาก ซึ่งในประเทศไทยนั้นยังถือว่ามีผู้รู้แล้วเข้าใจรายละเอียดทางด้านเทคโนโลยีการของระบบความปลอดภัยน้อยมาก วิทยานิพนธ์นี้จึงได้ศึกษาจัดทำเป็นหัวข้อเพื่อพัฒนาระบบรักษาความปลอดภัยทางโรงงานอุตสาหกรรมและอาคารสำนักงานของประเทศไทย โดยได้ศึกษาจัดทำเพื่ออำนวยความสะดวกและเป็นมาตรฐานในการออกแบบ การติดตั้งหัวล่อฟ้า เพื่อเกิดประโยชน์แก่บุคคลากรและระบบอุตสาหกรรมไทยให้มากที่สุด

LIGHTNING PROTECTION DESIGN PROGRAM ON WEB APPLICATION

Mr.Saksit maharat 46015374

Mr.Amnach Khawne Advisor

Acedemic Year 2006

Abstract

Thailand is rapidly growing in Technology, Economy and industry. With the increase in safe environmental awareness, both employee and organization require a safe work environment especially the system of computer. Thesis, themselves. Unfortunately, Thailand has only few experts in the technology of safety. Therefore, this paper aims to be the most useful for Thailand industries and office buildings in the development of safety by determining the standard of designing for the ligbtning protection system.

We found that people are looking for knowledge and do not understand to the importance of the safety system. We encourage to knowledge to promote more information regarding the industrial safety and protection system. There are many safety and protection systems such as CCTV, Surge protection, Fire alarm, Finger scanner, Key card access. This thesis proposes the lightning protection system

In this study , we used computer system for calculating and designing the lightning protection system, which makes more convenient and accurate for the users. Also, this system will be a guideline in developing the safety system in Thailand.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้อย่างดีด้วยคำแนะนำ คำปรึกษาและคอยดูแลจากหลายๆฝ่ายด้วยกัน โดยเฉพาะอาจารย์ที่ปรึกษาที่ให้โอกาสให้ข้าพเจ้าได้ทำปริญญานิพนธ์ฉบับนี้ คอยให้ความเอาใจใส่ แนะนำและให้ความช่วยเหลือเสมอมาคือ อาจารย์อำนาจ ขาวเน ซึ่งต้องขอขอบพระคุณเป็นอย่างสูง

ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์ที่ได้จัดเตรียมสิ่งอำนวยความสะดวก เพื่อให้การวิจัยและพัฒนาเป็นไปได้ด้วยความสะดวกและรวดเร็ว รวมทั้งยังมีอินเทอร์เน็ตความเร็วสูงให้บริการสำหรับการค้นคว้าหาความรู้ต่างๆ ซึ่งท้ายที่สุดแล้วก็ประกอบกันเป็นส่วนหนึ่งของโครงงานนี้

ขอขอบคุณพี่ๆ เพื่อนๆ น้องๆ ในห้องปฏิบัติการที่คอยสร้างความครึกครื้นยามอยู่ในห้องเป็นกำลังใจเสมอมาในการทำงาน

และสุดท้ายต้องขอขอบคุณบุคคลที่สำคัญที่สุดในชีวิตที่ทำให้ข้าพเจ้ามีวันนี้นั่นคือบิดา มารดา และบุคคลในครอบครัวอันเป็นที่เคารพรักซึ่งได้เลี้ยงดูคอยสั่งสอนข้าพเจ้ามาเป็นอย่างดี พร้อมให้โอกาสในการศึกษาอย่างเต็มที่และยังให้กำลังใจความรักเสมอมา ข้าพเจ้าขอกราบขอบพระคุณมา ณ ที่นี้ด้วย

นายศักดิ์สิทธิ์ มหาราต

สารบัญ

สารบัญ(ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป(ต่อ)

สารบัญรูป(ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาของปัญหา

ในโลกปัจจุบันนี้เทคโนโลยีนั้นเป็นปัจจัยที่มีผลสำคัญมากของระบบอุสาหกรรม ธุรกิจ หรือแม้กระทั้งการดำเนินชีวิตในสังคม ระบบอุตสาหกรรมในปัจจุบันมีการพัฒนาไปอย่างรวดเร็วและมีประสิทธิภาพ ในหลายหลายรูปแบบของระบบอุตสาหกรรมได้มีการนำเทคโนโลยีเข้ามาใช้แทนบุคลากรเพื่อให้ได้มาซึ่งความสะดวกรวดเร็วและมีประสิทธิภาพมากขึ้น เนื่องจากเทคโนโลยีในปัจจุบันนั้นมีความถูกต้องแม่นยำมากขึ้น ซึ่งสิ่งที่ได้รับการพัฒนาควบคู่ไปกับระบบอุตสาหกรรมก็คือระบบรักษาความปลอดภัยในโรงงานและอาคาร แม้กระทั่งรวมถึงตัวอุปกรณ์ต่างๆที่ใช้งานในระบบอุตสาหกรรมต่างๆ ซึ่งในส่วนนี้เป็นส่วนที่สำคัญแต่เนื่องด้วยการรักษาความปลอดภัยต่างๆของระบบอุตสาหกรรมและตัวอาคารนั้น ยังเป็นที่ไม่แพร่หลายนักในประเทศไทย แต่ระบบอุตสาหกรรมใหญ่หลายๆที่ก็เริ่มนำเข้ามาใช้เพื่อประสิทธิภาพที่ได้รับมากขึ้น แต่บุคลากรที่มีความรู้และศึกษาด้านนี้มีน้อยมาก เนื่องจากข้อมูลข่าวสารต่างๆยังมีให้ศึกษาเข้าใจน้อยมาก ซึ่งในกลุ่มคนส่วนมากยังมองเห็นว่ามิใช่เรื่องสำคัญใดๆ แต่ทางประเทศไทยก็ได้มีบัญญัติข้อกำหนดการรักษาความปลอดภัยให้เห็นหลายฉบับ ในส่วนระบบป้องกันอันตรายจากฟ้าผ่ามี กฎกระทรวง ฉบับที่ 33 (พ.ศ.2535)ออกตามความในพระราชบัญญัติควบคุมอาคาร พ.ศ.2522 ข้อที่ 13 รายละเอียดมีดังนี้

อาคารสูงต้องมีระบบป้องกันอันตรายจากฟ้าผ่า ซึ่งประกอบด้วย เสา ล่อฟ้า สายล่อฟ้า สายตัวนำสายนำลงดิน และหลักสายดินที่เชื่อมโยงกันเป็นระบบสำหรับ สายนำลงดินต้องมีขนาดพื้นที่ภาคตัดขวางเทียบได้ไม่น้อยกว่าสายทองแดง ตีเกลียว ขนาด 30 ตารางมิลลิเมตร สายนำลงดินนี้ต้องเป็นระบบที่แยกเป็นอิสระจากระบบสายดินอื่น อาคารแต่ละหลังต้องมีสายตัวนำโดยรอบอาคาร และมีสายนำลงดินต่อ จากสายตัวนำห่างกันทุกระยะไม่เกิน 30 เมตร วัดตามแนวของรอบอาคาร ทั้งนี้ สายนำลงดินของอาคารแต่ละหลังต้องมีไม่น้อยกว่าสองสายเหล็กเสริมหรือเหล็กรูปพรรณในโครงสร้างอาคารอาจใช้เป็นสายนำลงดินได้ แต่ต้องมีระบบการถ่ายประจุไฟฟ้าจากโครงสร้างสู่หลักสายดินได้ถูกต้องตามหลักวิชาการช่างระบบป้องกันอันตรายจากฟ้าผ่าให้เป็นไปตามมาตรฐานเพื่อความปลอดภัยทางไฟฟ้าของสำนักงานพลังงานแห่งชาติ

(ให้ไว้ ณ วันที่ 14 กุมภาพันธ์ พ.ศ. 2535 (ลงชื่อ) พลเอก อิสระพงศ์ หนุนภักดี รัฐมนตรีว่าการกระทรวงมหาดไทย)

ซึ่งในตอนนี้เราเล็งเห็นถึงปัญหาของการให้ความรู้ความเข้าใจในส่วนของระบบรักษาความปลอดภัยนั้นน้อยเกินไป เราจึงได้เห็นว่าน่าจะมีสื่อวิธีให้ความเข้าใจที่เป็นมาตรฐานในการวางระบบรักษาความปลอดภัยในระบบอุตสาหกรรม ซึ่งระบบรักษาความปลอดภัยในระบบอุตสาหกรรมนี้มี

มากมาย เช่น ระบบทีวีวงจรปิด ระบบป้องกันไฟฟ้ากระชาก ระบบเตือนอัคคีภัย ระบบสแกนลายนิ้วมือ และระบบคีย์การ์ดฯลฯ ในส่วนนี้ขอยกตัวอย่างระบบป้องกันอันตรายจากฟ้าผ่าของบริษัทซันเทคเทคโนโลยี มาเป็นแนวทางของการนำเสนอเนื่องจากเป็นระบบที่มีคนเข้าใจและได้รับข่าวสารน้อย และเป็นสิ่งที่ปัจจุบันได้เข้ามีบทบาทในระบบอุตสาหกรรมของประเทศไทยมากขึ้น และทางบริษัทยินดีให้ความร่วมมือในการค้นคว้าให้ข้อมูลต่างๆ

โดยในการศึกษาค้นคว้าได้นำคอมพิวเตอร์เข้ามามีบทบาทในการคำนวณวางแบบระบบป้องกันภัยอันตรายจากฟ้าผ่า เพื่อให้เกิดความสะดวก มาตรฐานความถูกต้องแม่นยำแก่ผู้ที่ใช้เทคโนโลยีออกแบบระบบป้องกันภัยอันตรายจากฟ้าผ่า และเป็นแนวทางในการพัฒนาระบบอุตสาหกรรมของประเทศไทยให้ก้าวหน้ามากขึ้น

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อวิเคราะห์และพัฒนาระบบรักษาความปลอดภัยในระบบอุตสาหกรรมและตัวอาคารสถานที่ โดยได้นำเทคโนโลยีต่างๆมาใช้

1.2.2 ศึกษาเกี่ยวกับการทำงานของภาษา JAVASCRIPT XMLHttpRequest AJAX PHP เพื่อนำมาใช้ในการเขียนโปรแกรมบนเว็บแบบพลิเคชั่น โดยให้โปรแกรมสามารถใช้งานได้ในระบบ Network ที่ใช้งานอยู่

1.2.2 ศึกษาเกี่ยวกับการทำงานของระบบป้องกันภัยอันตรายจากฟ้าผ่าเพื่อนำข้อมูลที่ได้มาใช้ในการออกแบบและพัฒนาโปรแกรมออกแบบระบบติดตั้งหัวล่อฟ้า บนเว็บแอพพลิเคชั่น

1.2.4 ศึกษาการทำงานของเซิร์ฟเวอร์ ของบริษัทเพื่อนำมาพัฒนาโปรแกรมเพื่อให้สามารถใช้งานบนเว็บแอพพิลเคชั่นของบริษัทที่มีอยู่นั้นได้จริง

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1.3.1 ได้รับความรู้ในส่วนของระบบป้องกันภัยอันตรายจากฟ้าผ่า และระบบต่างๆที่เกี่ยวข้อง

1.3.2 ลดค่าใช้จ่ายในการนำเข้าของโปรแกรม

1.3.3 มีความรู้ความสามารถในการเขียนโปรแกรมภาษา JAVASCRIPT XMLHttpRequest AJAX PHP เพื่อนำมาใช้ในการเขียนโปรแกรมบนเว็บแบบพลิเคชั่น

1.3.4 ได้โปรแกรมที่สามารถนำมาใช้เป็นมาตราฐานในการออกแบบหัวล่อฟ้า

1.3.5 ได้นำเทคโนโลยีที่น่าสนใจ มาใช้ให้เป็นประโยชน์สูงสุด

1.4 ขอบเขตของโครงงาน

1.4.1 ออกแบบระบบป้องกันภัยอันตรายจากฟ้าผ่าได้

1.4.2 เพื่อให้สามารถพัฒนาโปรแกรมระบบป้องกันภัยอันตรายจากฟ้าผ่าได้

1.4.3 สามารถทำให้ระบบป้องกันภัยอันตรายจากฟ้าผ่า สามารถใช้งานได้บนเว็บแอพพลิเคชั่น และบนเซิร์ฟเวอร์ที่มีอยู่นั้นได้โดยปกติ

1.5 ส่วนประกอบของรายงาน

รายงานฉบับนี้ได้แบ่งเนื้อหาออกเป็น 5 บทด้วยกันคือ

บทที่ 1 ความเป็นมาของปัญหา วัตถุประสงค์ของโครงงาน ประโยชน์ที่คาดว่าจะได้รับ ขอบเขตของโครงงาน และส่วนประกอบของปริญญานิพนธ์

บทที่ 2 ทฤษฎีพื้นฐานที่ใช้ในโครงงาน ซึ่งประกอบด้วยทฤษฎี 5 ทฤษฎีหลักๆคือ

1. ระบบป้องกันภัยอันตรายจากฟ้าผ่า
2. การทำงาน Web Proramming ด้วย Ajax
3. ภาษา XMLHttpRequest
4. ภาษา JavaScript
5. ภาษา PHP

บทที่ 3 ได้อธิบายถึงการออกแบบระบบ 2 หัวข้อด้วยกันได้แก่

1. แนวทางการออกแบบระบบ
2. ขั้นตอนและวิธีการออกแบบระบบ

บทที่ 4 การทดลองและผลการทดลอง

บทที่ 5 บทวิจารณ์และสรุป ซึ่งกล่าวถึง 4 ส่วนใหญ่ๆ คือ

1. บทสรุป
2. ปัญหาที่ได้พบระหว่างดำเนินโครงงาน
3. แนวทางการพัฒนาในอนาคต

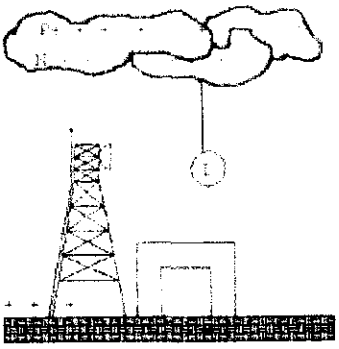
บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 แนวคิดเรื่องการป้องกันฟ้าผ่าแบบใหม่

นานมาแล้วที่มนุษย์พยายามควบคุมภัยจากธรรมชาติและหนึ่งในหลายๆ เรื่องที่นักวิทยาศาสตร์พยายามทำมาอย่างต่อเนื่องคือการควบคุมฟ้าผ่า นักวิจัยต่างก็พยายามหาวิธีป้องกันที่มีประสิทธิภาพสูงสุด ด้วยงานวิจัยที่มีมาอย่างต่อเนื่องนี้เอง จึงได้ผลิตภัณฑ์หลายระบบที่อ้างว่าสามารถป้องกันได้ดีกว่าใช้เทคโนโลยีที่สูงกว่าและที่สำคัญต่างก็อ้างว่าดีกว่าสินค้าคู่แข่ง ความสำเร็จแรกคือระบบพื้นฐาน Franklin-rod system ที่คิดค้นโดย เบนจามิน แฟรงคลิน (Benjamin Franklin) ในปี พ.ศ. 2295 [3] ซึ่งเป็นระบบที่อยู่บนพื้นฐานที่ว่าถ้าฟ้าผ่าลงมาก็ทำให้มันลงดินเสียก่อนที่มันจะทำลายสิ่งอื่นๆ อย่างไรก็ตามระบบนี้ก็ยังไม่สามารถหยุดยั้งการเกิดฟ้าผ่าได้ หลังจากนั้น นักประดิษฐ์ได้คิดค้นวิธีอื่นๆ อีกหลายวิธีรวมทั้งวิธีที่มีพื้นฐานเดียวกันกับวิธีแฟรงคลิน เช่น วิธี Early streamer Emission (ESE) วิธีกัมมันตรังสีวิธีใช้แบตเตอรี่ ยิ่งกว่านั้นบางวิธีสามารถใส่ประจุไฟฟ้าไม่ให้ฟ้าผ่าลงมาที่สิ่งปลูกสร้างหรือยับยั้งก่อนจะเกิดฟ้าผ่าเลยก็มี และเพราะการแข่งขันมีสูงขึ้นเรื่อยๆ นี้เองทำให้การวิจัยเรื่องนี้เป็นไปอย่างต่อเนื่องและจริงจัง ผลการวิจัยจึงน่าติดตามและน่าสนใจอย่างยิ่งจากเหตุผลที่กล่าวมานั้น จำเป็นอย่างยิ่งที่จะต้องมีหน่วยงานที่เป็นกลางมาทำหน้าที่คอยตรวจสอบสินค้าที่กล่าวอ้างข้างต้น หน่วยงานดังกล่าวเช่น International Electrotechnique Commission(IEC) ผลการตรวจสอบน่าสนใจยิ่ง จึงนำรายละเอียดบางส่วนมาเผยแพร่ในบทความนี้ก่อนอื่นเราจะพิจารณากระบวนการเกิดฟ้าผ่าแบบคร่าวๆ กันก่อน เริ่มจากมีประจุต่างชนิดกันระหว่างกลุ่มเมฆกับพื้นโลกโดยมีอากาศเป็นฉนวนคั่น ซึ่งในภาวะปกติอากาศจะไม่นำไฟฟ้า แต่เมื่อมีสนามไฟฟ้าอันเนื่องจากประจุไฟฟ้าบนเมฆดังกล่าวสูงมากพอ จะทำให้อากาศเริ่มแตกตัวเป็นอิออน และงอกเป็น “งวง” ขึ้นออกจากเมฆสู่พื้นโดยงวงดังกล่าวจะงอกแล้วหยุดเป็นช่วงๆ เรียกว่า Stepped Leaderซึ่งงวงอิออนนี่เองที่เป็นทางนำกระแสไฟฟ้าได้ และเมื่อทางนำดังกล่าว “งอก” จนใกล้จะถึงพื้นก็จะมีลำประจุชนิดตรงข้ามงอกขึ้นมาบรรจบกันเรียกว่า Streamer เมื่อเชื่อมกันแล้วจะเกิดการถ่ายเทประจุระหว่างกันอย่างมโหฬารเรียกว่า Return Stroke

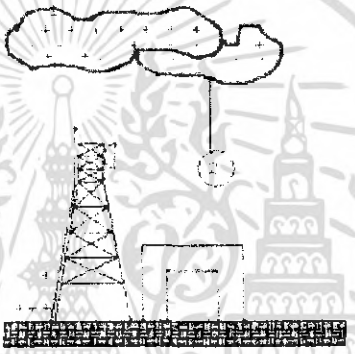
2.1.1 การเกิดฟ้าผ่า

"ฟ้าผ่า" เป็นปรากฏการณ์ที่เกิดขึ้นตามธรรมชาติโดยเริ่มจากการก่อตัวของเมฆฟ้าผ่า (Cumulonimbus Cloud) ที่มีทั้งประจุบวกและลบอยู่ในก้อนเมฆ เมื่อการสะสมประจุมากขึ้นก็ทำให้ศักดาไฟฟ้าระหว่างก้อนเมฆกับพื้นดินมีการพัฒนา เพิ่มสูงขึ้นจนถึงจุดสูงสุดที่ทำให้เกิดการถ่ายเทประจุไฟฟ้าปริมาณมหาศาลระหว่างก้อนเมฆกับพื้นดินที่เรียกว่าฟ้าผ่า กระบวนการดังกล่าวมีขั้นตอนดังนี้คือ



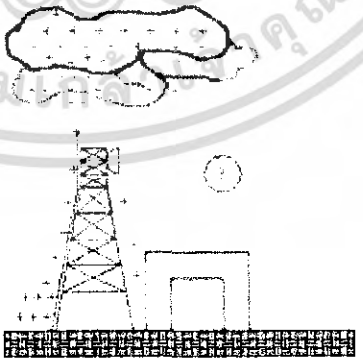
รูปที่ 2.1 แสดงการเกิดฟ้าผ่า Step ที่ 1

1. เริ่มก่อตัวของประจุไฟฟ้าทั้งประจุบวก (P) และประจุลบ (N) ภายในก้อนเมฆฟ้าผ่า



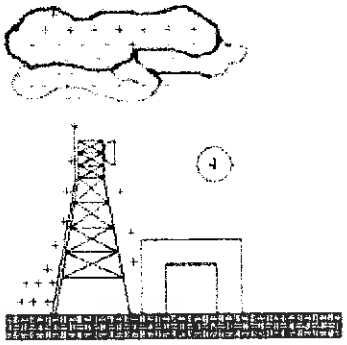
รูปที่ 2.2 แสดงการเกิดฟ้าผ่า Step ที่ 2

2. การถ่ายเทประจุบวกและลบภายในก้อนเมฆชั้นต่างๆ โดยชั้นที่ ไม่เกิดความแปรปรวนจะแสดงศักย์ไฟฟ้าเป็นบวก และชั้นที่อยู่ต่ำ เกิดความแปรปรวนจะแสดงศักย์ไฟฟ้าเป็นลบและเคลื่อนตัวต่ำลงตามแรงดึงดูดของโลก



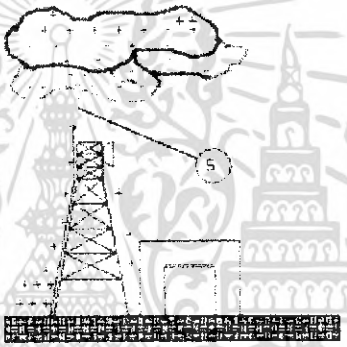
รูปที่ 2.3 แสดงการเกิดฟ้าผ่า Step ที่3

3. ที่ฐานก้อนเมฆแสดงศักย์ไฟฟ้าเป็นลบเคลื่อน ตัวต่ำลงสู่พื้นดินที่มีศักย์ไฟฟ้าเป็นบวก มากกว่า



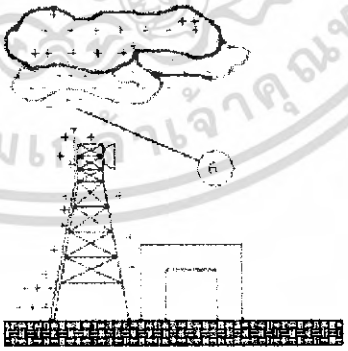
รูปที่ 2.4 แสดงการเกิดฟ้าผ่า Step ที่ 4

4. เมื่อก้อนเมฆเคลื่อนตัวลงต่ำทำให้ความต่างศักย์ระหว่างก้อนเมฆ กับพื้นดินเพิ่มสูงขึ้น



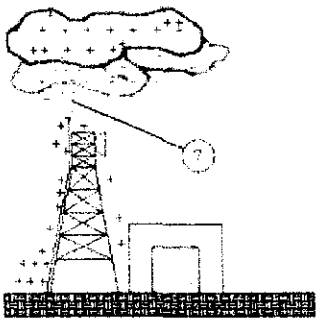
รูปที่ 2.5 แสดงการเกิดฟ้าผ่า Step ที่ 5

5. เกิด step leader ขึ้น มีศักย์ไฟฟ้าเป็นลบ เคลื่อนที่ลงสู่พื้นดินที่มีศักย์ไฟฟ้าเป็นบวก



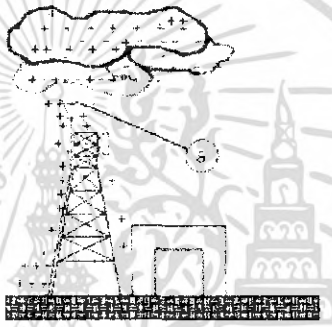
รูปที่ 2.6 แสดงการเกิดฟ้าผ่า Step ที่ 6

6. เกิด upward streamers ขึ้น มีศักย์ไฟฟ้าเป็นบวกเคลื่อนที่ เข้าหา step leader ที่มีศักย์ไฟฟ้าเป็นลบ



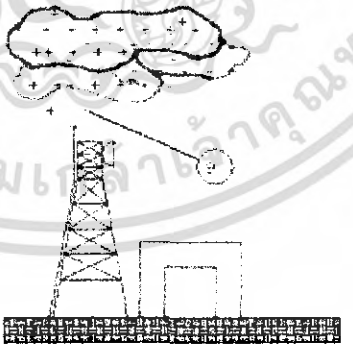
รูปที่ 2.7 แสดงการเกิดฟ้าผ่า Step ที่ 7

7. step leader เคลื่อนที่ชนกับ upward streamers เกิด lightning channel current ขึ้นและกระแสจะเริ่มไหล



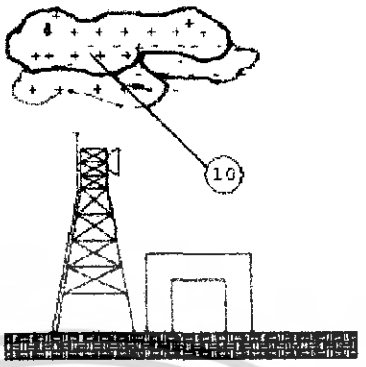
รูปที่ 2.8 แสดงการเกิดฟ้าผ่า Step ที่ 8

8. ประจุบวกที่พื้นดินซึ่งมีจำนวนมากเคลื่อนที่ขึ้นสู่ก้อนเมฆที่มีประจุ บวกน้อยกว่าเรียกกระบวนการนี้ว่า return stroke ซึ่งจะมี กระแสไหล



รูปที่ 2.9 แสดงการเกิดฟ้าผ่า Step ที่ 9

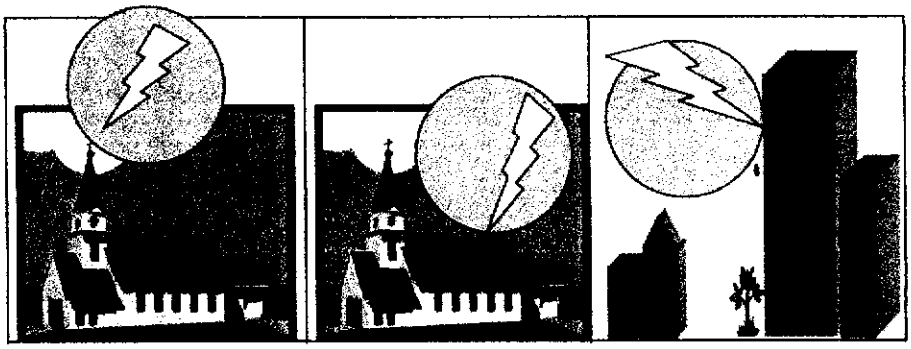
9. ศักย์ไฟฟ้าบริเวณฐานก้อนเมฆพยายามถ่าย ประจุ เพื่อกลับสู่สภาวะสมดุลเรียกกระบวนการ นี้ว่า J & K phenomena



รูปที่ 2.10 แสดงการเกิดฟ้าผ่า Step ที่ 10

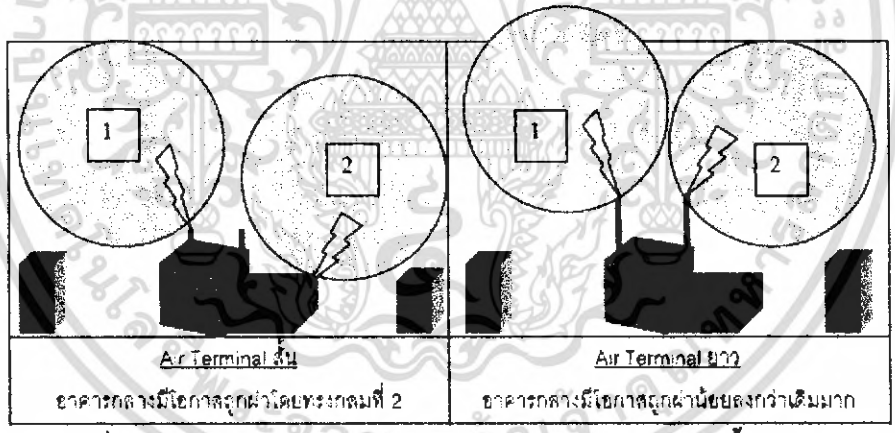
10. ศักย์ไฟฟ้าลบที่อยู่สูงกว่า ส่งถ่ายประจุลบมายัง ฐานก้อนเมฆ ซึ่งแสดงศักย์ไฟฟ้าเป็นบวกมากกว่า เกิดเป็นลำแสงเรียกว่า Dart leader ถ้าการส่งถ่ายยังเหลือศักย์ไฟฟ้าลบอยู่บริเวณฐานก้อน เมฆมีปริมาณมากเมื่อเทียบกับพื้นดินจะทำให้เกิด ฟ้าผ่าซ้ำได้จากรูปจะพบว่าขั้นตอนที่ 8 จะมีกระแสฟ้าผ่าไหลสูงสุด ซึ่งเหมาะสมที่จะวัดค่ากระแสและนำมากำหนดค่าความต้านทาน ระหว่างแท่งกราวด์กับ remote earth เพื่อใช้ในการออกแบบระบบกราวด์ของระบบล่อฟ้าต่อไป

ส่วนจุดสำคัญที่จะนำมาใช้ในการป้องกันคือ ในขั้นตอน Stepped Leader ขั้นสุดท้ายก่อนที่จะเกิดการเชื่อมต่อกัน หรือก่อนจะเกิด Return Stroke นั้น นักประดิษฐ์ได้ใช้ประโยชน์จากจังหวะนี้มาสร้างระบบป้องกัน โดยต่อตัวนำยื่นออกไปในอากาศเรียกว่าตัวนำล่อฟ้าหรือสายล่อฟ้า(air terminal) มารอรับประจุไฟฟ้าจากส่วนที่สูงที่สุดของอาคารหรือหลังคาสิ่งปลูกสร้าง เพื่อให้ Return Stroke เกิดขึ้นที่ปลายของตัวนำที่สร้างขึ้นนี้ โดยมีความสำคัญอย่างยิ่งที่จะต้องจัดให้ตัวนำล่อฟ้านั้นให้อยู่สูงกว่าผิวของอาคารที่ต้องการป้องกันมิเช่นนั้นแล้ว Return Stroke จะเกิดในจุดอื่นที่ไม่พึงประสงค์ซึ่งจะก่อให้เกิดความเสียหายต่อส่วนนั้นๆ ได้ด้วยการพิจารณาแบบนี้เราพบว่าไม่เพียงแค่ดาดฟ้าตึกเท่านั้นที่มีโอกาสเกิด Return Stroke หากยังรวมไปถึงด้านข้างตึกก็มีโอกาสเช่นกัน แนวคิดในการป้องกันนั้นให้คิดเสมือนว่า Stepped Leader งอกลงมาใกล้จะถึงพื้นแล้วและแน่นอนว่าจะเกิด Return Stroke ในจุดที่ใกล้กับ Stepped Leader มากที่สุดโดยไม่คำนึงว่าจุดนั้นจะเป็นตัวนำหรือฉนวน โดยนัยนี้การป้องกันจึงคิดเสมือนว่ามีทรงกลมขนาดยักษ์ [2]กลิ้งลงมาจากฟ้าและไม่จำกัดว่าจะหล่นในแนวดิ่งหรือไม่ หรืออีกนัยหนึ่งทรงกลมนี้ถูกลมพัดเฉียงไปด้านข้างได้ด้วย และที่แน่นอนที่สุดคือเมื่อผิวของทรงกลมสัมผัสกับสิ่งใด สิ่งนั้นมีโอกาสจะถูกฟ้าผ่าได้การป้องกันจึงจำต้องจัดให้ทรงกลมกลิ้งดังกล่าว ไม่ให้สัมผัสกับสิ่งอื่นเลยนอกจากตัวนำล่อฟ้าที่สร้างขึ้นเท่านั้น การป้องกันจึงจะถือว่าใช้ได้ ปัญหาจึงมีเพิ่มมาอีกว่าทรงกลมดังกล่าวมีขนาดเท่าใด คำตอบก็คือใช้ขนาดรัศมี 20-30 เมตรก็เพียงพอแล้ว ดูรูปที่ 2.11 ประกอบ



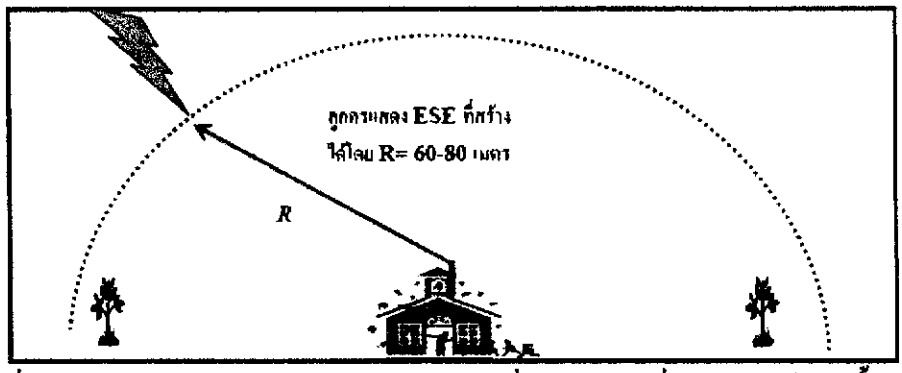
รูปที่ 2.11 จุดต่างๆ ที่มีโอกาสถูกฟ้าผ่าตามหลักทรงกลมกลิ้งโดยจุดที่ถูกผ่าคือจุดที่ผิวของทรงกลมสัมผัสถึงได้

ตามหลักการนี้ถ้าต่อตัวนำล่อฟ้าให้ยื่นออกไปได้มากเท่าไหร่ยิ่งป้องกันได้ดียิ่งขึ้นดังแสดงในรูปที่ 2.12 นักประดิษฐ์ได้คิดค้นวิธีที่จะต่อตัวนำล่อฟ้านั้น ให้ยาวขึ้นไปอีกแต่แทนที่จะต่อกันจริงๆ กลับใช้วิธี เสมือนจริง กล่าวคือ เมื่อเกิดความเครียดของสนามไฟฟ้าสูงที่ตัวนำล่อฟ้านั้น อุปกรณ์ป้องกันซึ่งประดิษฐ์ขึ้นใหม่ที่ประกอบอยู่ภายใน จะปล่อยประจุอิสระออกมาเรียกว่า Early Streamer Emission :ESE และประจุอิสระนี้ก็จะวิ่งเข้าสู่ Stepped Leader ก่อนที่จะเกิดขึ้นเองตามธรรมชาติเสียอีก นั่นหมายความว่าก่อนที่จะเกิด Return Stroke ที่จุดอื่นนั้น อุปกรณ์ ESE ชิงตัดหน้าสร้างทางนำประจุขึ้นมาแล้วและขึ้นออกไปสู่ Stepped Leader ซึ่งเป็นเหตุให้เกิด Return Stroke ในเสี้ยววินาทีถัดมา ตามทางนำที่เกิดขึ้นนี้ จากนั้นจึงนำประจุลงสู่พื้นดินต่อไปโดยผ่านทางตัวนำที่ต่อไว้แล้ว



รูปที่ 2.12 แสดงการเปรียบเทียบการป้องกันของ Air Terminal สั้นและยาว

ด้วยแนวคิดนี้สินค้าบางรายอ้างว่าสามารถสร้าง Early Streamer Emission ได้ยาวถึง 60-80 เมตร นั่นหมายความว่าเมื่อติดตั้งตัวนำล่อฟ้าแบบนี้แล้วจะเสมือนว่ามีตัวนำยื่นออกมาในอากาศ ยาวถึง 60-80 เมตร และชี้ออกในแนวรัศมีทุกทิศทาง การป้องกันจึงเป็นดังรูปที่ 2.13



รูปที่ 2.13 แสดงแนวคิดของการป้องกันแบบ ESE ซึ่งฟ้าจะผ่าจุดที่เตรียมไว้แล้วเท่านั้น โดยจุดอื่นๆ ในรัศมี R จะไม่ถูกฟ้าผ่าเลย

2.1.2 สูตรการคำนวณหาค่ารัศมีป้องกันฟ้าผ่าของหัวล่อฟ้า Stormaster ESE terminal

จากห้องทดลอง ซึ่งได้มาตราฐาน Standard NF C 17-102 (July 1995) จากประเทศฝรั่งเศส.

$Rp = \sqrt{h(2D-h)+\Delta T(2D+\Delta T)}$ เมื่อความสูงเสามากกว่า 5 เมตร

• เมื่อ ΔT คือค่าการส่งผ่านกระแสไฟฟ้าในอากาศ

Stormaster-ESE-50 = จะมีค่า ΔT (μs)50

Stormaster-ESE-60 = จะมีค่า ΔT (μs) 60

• h = คือค่าความสูงของเสาหัวล่อฟ้าที่บริเวณที่ต้องการได้รับการป้องกันจากอันตรายจากฟ้าผ่า(มีค่าเป็นเมตร)

• D = คือค่าตัวแปลคงที่ ซึ่งจะมีค่าขึ้นอยู่กับ Level ที่เลือกมาคำนวณ

ซึ่ง ค่าค่านี้ได้รับมาตรฐานของ standard NF C 17-102.

D = 20m for protection level 1 (High Protection)

D = 45m for protection level 2 (Medium protection)

D = 60m for protection level 3 (Standard protection)

PROTECTION RADIUS (M) - (Rp)

PROTECTION RADIUS (M) - (Rp)									
h = height of Stormaster terminal above area to be protected (m)	2	4	5	6	10	15	20	45	60
Protection Level 1 (High Protection)									
Stormaster 50	28	55	68	69	69	70	70	-	-
Stormaster 60	32	64	79	79	79	80	80	-	-
Protection Level 2 (Medium Protection)									
Stormaster 50	35	69	86	87	88	90	92	95	-
Stormaster 60	40	78	97	97	99	101	102	105	
Protection Level 3 (Standard Protection)									
Stormaster 50	38	76	95	96	98	100	102	110	110
Stormaster 60	44	87	107	107	109	111	113	120	120

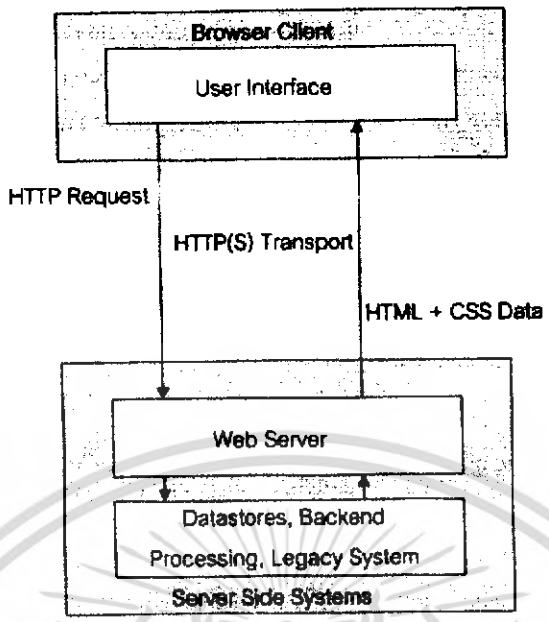
รูปที่ 2.14 ตารางข้อมูลรัศมีหัวล่อฟ้า

2.2 ทำความรู้จักกับ AJAX

ปัจจุบันเทคโนโลยีต่างๆที่ใช้กับเครือข่ายอินเตอร์เน็ตมีการเปลี่ยนแปลงไปอย่างรวดเร็ว จากยุคแรกอินเตอร์เน็ตถูกพัฒนาเพื่อใช้ในการแลกเปลี่ยนข่าวสารข้อมูลเท่านั้น แต่ในปัจจุบันได้มีการพัฒนาเทคโนโลยีใหม่ขึ้นมาอีกมากมายเพื่อทำงานบนเครือข่ายอินเตอร์เน็ต จนทำให้อินเตอร์เน็ตกลายมาเป็นศูนย์กลางในการเผยแพร่ข่าวสารข้อมูล และที่สำคัญกลายมาเป็นช่องทางในการทำธุรกิจอีกด้วย ดังจะเห็นได้จากเว็บไซต์ขายสินค้าจำนวนมาก ซึ่งเว็บไซต์เหล่านั้น ก็คือเว็บแอปพลิเคชั่นที่เป็นเทคโนโลยีหนึ่งบนเครือข่ายอินเตอร์เน็ตที่ได้รับความนิยมอย่างแพร่หลายมาจนถึงปัจจุบัน นับตั้งแต่อดีต กลไกการทำงานของเว็บแอปพลิเคชั่นได้ถูกพัฒนาให้มีประสิทธิภาพมากขึ้นเรื่อยๆเพื่อให้สามารถรองรับการแข่งขันทางธุรกิจที่เพิ่มมากขึ้นได้ จนถึงปัจจุบัน AJAX เป็นแนวคิดหนึ่งที่ทำให้เว็บแอปพลิเคชั่นมีประสิทธิภาพมากขึ้น สามารถตอบสนองความต้องการของผู้ใช้ต่อไปในอนาคตได้

2.2.1 ความหมายของ AJAX

AJAX ย่อมาจากคำว่า "Asynchronous JavaScript And XML" เป็นการนำเอาเทคโนโลยีต่างๆมาทำงานร่วมกัน หรือกล่าวได้อีกนัยหนึ่งว่า AJAX ไม่ใช่เทคโนโลยีใหม่ แต่เป็นเพียงเทคนิคการทำงานร่วมกันของหลายๆเทคโนโลยีอันได้แก่ DHTML,CCS,JavaScript,DOM,XML และ XMLHttpRequest Object ผู้ที่ริเริ่มเทคนิคนี้คือ Jesse JamesGarrett ซึ่งเห็นว่าจำนวนผู้ใช้เว็บแอปพลิเคชั่นมีมากขึ้นเรื่อยๆ ดังนั้นเพื่อที่จะตอบสนองการให้บริการกับผู้ใช้ที่มีจำนวนมากให้สามารถใช้เว็บแอปพลิเคชั่นได้เร็วใกล้เคียงกับการใช้แอปพลิเคชั่นทั่วไปที่ไม่ได้ทำงานบนเว็บ เช่น โปรแกรมพิมพ์เอกสาร(Word Processor) เป็นต้น โดยให้โต้ตอบกับ User ได้ทันทีและใช้เวลาน้อยที่สุด จึงได้พัฒนาเทคนิคที่เรียกว่า "AJAX" ขึ้นมา ความสามารถของ AJAX นั้นทำให้เว็บแอปพลิเคชั่นแสดงข้อมูลบนเว็บเพจ(Web Page) ได้อย่างรวดเร็วและมีความนุ่มนวล เนื่องจากแสดงผลเพียงบางส่วนบนหน้าจอ ซึ่งเป็นบริเวณที่ User ต้องการข้อมูลจริงๆเท่านั้น



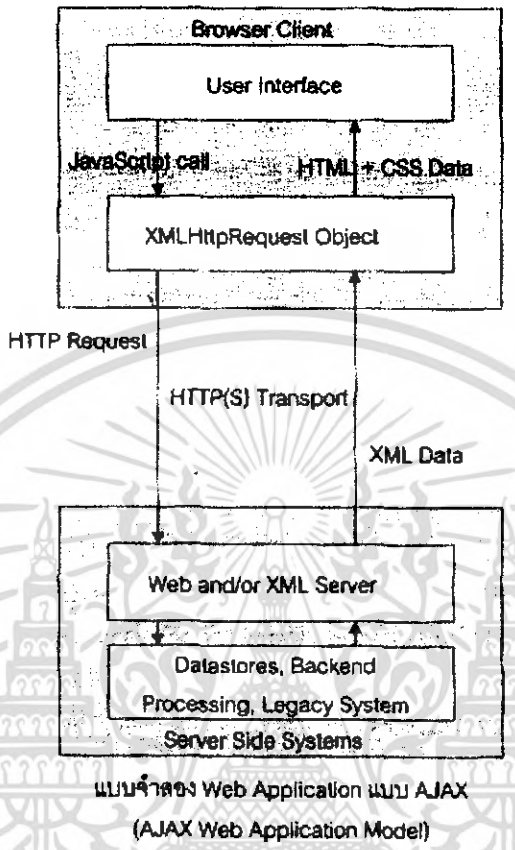
แบบจำลอง Web Application ทั่วไป
(Classic Web Application Model)

รูปที่2.15 แสดงแบบจำลองการทำงานของเว็บแอปพลิเคชั่นแบบเดิม

การทำงานของเว็บแอปพลิเคชั่นแบบเดิม(Classic Web Application)จะเริ่มจากผู้ใช้(User)เปิดเว็บบราวเซอร์(Web Browser) เช่น Internet Explorer(IE)หรือ Firefox แล้วร้องขอข้อมูลหรือเว็บเพจ(Web Page) ที่เราต้องการ โดยการพิมพ์ชื่อ ที่อยู่ หรือที่เราเรียกว่า "URL" เช่น www.ce.kmitl.ac.th หรือ www.suntec.co.th เป็นต้น สำหรับ User ที่เปิด Web Browser ก็คือ ลูกค้า หรือ ผู้ใช้ ที่อยู่ทางฝั่งไคลเอนต์(Client Side) จากนั้น URL ที่ถูกร้องขอจะถูกส่งผ่านทางอินเตอร์เน็ตไปยังฝั่งผู้ให้บริการ (Server Side) เมื่อผู้ให้บริการได้รับการร้องขอก็จะจัดส่งเว็บเพจกลับมาให้ และทันทีที่ Server ส่งเว็บเพจมาตามที่ User ขอร้องไป การติดต่อระหว่างฝั่ง Server กับ Client จะสิ้นสุดหรือขาดการติดต่อจากกันทันที (ภาษาเทคนิคเรียกว่า"Stateless") ดังนั้นเมื่อ User ต้องการข้อมูลหรือเว็บเพจใหม่จาก Server เดิมต้องเริ่มการติดต่อหรือร้องขอ URL ใหม่อีกครั้ง Server จะส่งหน้าเว็บเพจใหม่มาให้ จากนั้นWeb Browser จะนำเว็บเพจที่ได้มานั้นแสดงเป็นหน้าใหม่ ผลที่เกิดขึ้นนั้นคือ จอกระพิบ เนื่องจากมีเว็บเพจใหม่จึงต้องรีเฟรช(Refresh)หน้าจอใหม่ มีความล่าช้าเนื่องจากต้องรอการประมวลผลที่ฝั่ง Server ให้เสร็จก่อนแล้วจึงส่งผลตอบกลับมาให้ User อีกทั้งข้อมูลที่ใช้ในการส่งแต่ละครั้งมีจำนวนมากทำให้ใช้แบนด์วิธ(Bandwidth) ค่อนข้างสูง ส่งผลให้การส่งข้อมูลเกิดความล่าช้าตามไปด้วย

จากปัญหาที่เกิดขึ้นกับเว็บแอปพลิเคชั่นแบบเดิม จึงเกิดแนวคิดใหม่ที่นำมาใช้ในการแก้ปัญหาดังกล่าว นั้นคือการใช้ AJAX โดยเว็บแอปพลิเคชั่นที่ใช้ AJAX สามารถลดการรีเฟรชของหน้าจอทำการการแสดงผลมีความนุ่มนวล และเปลี่ยนรูปแบบเอกสารให้มีขนาดเล็กลง โดยจากเดิมที่ส่งเป็นเว็บเพจ

ทั้งหน้าที่มีข้อมูลรวมกันมากมาย(HTML+CSS Data ในรูปที่ 2.15) กลายมาเป็นเพียงข้อมูลชิ้นเล็กๆ (XML Data ในรูป 2.16) ซึ่งใช้แบนด์วิธน้อยกว่าการส่งข้อมูลในเว็บแอปพลิเคชั่นแบบเดิม มีผลทำให้สามารถส่งข้อมูลได้เร็วขึ้น



รูปที่ 2.16 แสดงแบบจำลองการทำงานของเว็บแอปพลิเคชั่นแบบ AJAX

จากรูปที่ 2.16 หัวใจสำคัญของการทำงานของ AJAX คือ "XMLHttpRequest Object" ซึ่งทำหน้าที่ติดต่อร้องขอข้อมูลจากฝั่ง Server โดยข้อมูลที่ร้องขอนั้นเป็นข้อมูลเล็กๆทาง Server จัดหาข้อมูลและตอบกลับมาในรูปแบบ XML Data จากนั้นเป็นหน้าที่ของ JavaScript เดิมที่ปรากฏอยู่บน Web Browser การแสดงผลบนหน้าจอจะแสดงข้อมูลในช่องว่างหรือบางส่วนบนหน้าจอเท่านั้น ไม่ต้องรีเฟรชหน้าจอใหม่ อีกทั้งสามารถแสดงผลข้อมูลในเว็บเพจได้เร็วขึ้นเนื่องจากข้อมูลที่ส่งมาจากฝั่ง Server มีจำนวนน้อยไม่ต้องส่งเว็บเพจทั้งหน้าทำให้ใช้แบนด์วิธน้อยลง เช่น หาก User ป้อนข้อมูลที่เป็น "รหัสไปรษณีย์" ลงไปบนเว็บเพจข้อมูลที่เป็นส่วน "เมือง" หรือ "เขต" จะปรากฏขึ้นมาโดยทันทีโดยที่ไม่ต้องมีการรีเฟรชหน้าจอ

ดังนั้นลักษณะการทำงานของเว็บแอปพลิเคชั่นที่ทำงานตามแบบ AJAX จะคล้ายกับการทำงานของโปรแกรมทั่วไปที่ทำงานบน **Desktop** หรือทำงานบนระบบปฏิบัติการ **Windows** กล่าวคือ โปรแกรมจะได้ตอบกับผู้ใช้ได้ทันทีในเหตุการณ์ใดเหตุการณ์หนึ่งโดยไม่ต้องรอ

2.2.2 องค์ประกอบของ Ajax

ตามที่กล่าวมาก่อนแล้วว่า Ajax เป็นเทคนิคที่นำเอาหลายๆเทคโนโลยีมาทำงานร่วมกัน แต่ละเทคโนโลยีจะมีหน้าที่การทำงานที่ต่างกัน ดังนี้

JavaScript

JavaScript เป็นภาษายุคใหม่ถูกพัฒนาขึ้นโดย บริษัท Netscape Communications Corporation ซึ่งเป็นภาษาสคริปต์ที่นำมาใช้ในระบบอินเตอร์เน็ตเพื่อใช้ในการพัฒนาเว็บเพจต่างๆ ภาษา JavaScript เป็นภาษาที่มีความสามารถสูง เช่น สามารถเพิ่มลูกเล่นต่างๆให้กับเว็บเพจ และต็ตอบกับผู้ชมเว็บเพจได้ จึงนิยมนำมาสร้างเว็บเพจเพื่อให้ได้เว็บเพจที่มีศักยภาพมากขึ้น

สำหรับเว็บแอพพลิเคชั่นที่สร้างจาก Ajax จะใช้ JavaScript ควบคุมการแสดงข้อมูล รวมทั้งโต้ตอบกับ User กล่าวได้ว่าการทำงานหลักๆที่เกี่ยวข้องกับการตอบสนอง User จะเป็นหน้าที่ของ JavaScript ทั้งสิ้น

Cascading Style Sheet (CSS)

CSS เป็นภาษาที่ใช้กำหนดโครงสร้างหรือลักษณะการแสดงผลของเว็บเพจ ที่มีความคล้ายคลึงกัน เช่น สีของตัวอักษร ขนาดของข้อความที่เป็นหัวข้อ และสีของจุดเชื่อมโยง เป็นต้น ถ้าต้องการสร้างเว็บเพจเป็นจำนวนมากอาจทำให้เสียเวลาในการกำหนดลักษณะที่ซ้ำๆกันของเว็บเพจเหล่านี้ เราสามารถลดระยะเวลาลงได้ด้วยการกำหนดสไตล์ชีท ซึ่งก็คือ การกำหนดรูปแบบที่ต้องการให้กับเว็บเพจไว้ก่อน จากนั้นถ้าต้องการกำหนดรูปแบบ ณ ตำแหน่งใดในเว็บเพจ ก็ให้เรียกใช้งานสไตล์ชีทที่ได้กำหนดไว้ ทำให้เว็บเพจที่มีการเรียกใช้สไตล์ชีทเดียวกันจะมีลักษณะที่เหมือนกัน ซึ่งเป็นผลให้การสร้างและการแก้ไขเว็บเพจจะทำได้สะดวกรวดเร็วยิ่งขึ้น ด้วยความสามารถนี้เอง Ajax จึงนำ CSS มาใช้กำหนดโครงสร้างหรือลักษณะของการแสดงผลของเว็บเพจ ซึ่งนอกจากจะทำให้เว็บเพจ ดูน่าสนใจแล้วยังทำให้สามารถเรียกใช้รูปแบบต่างๆของสไตล์ชีทที่กำหนดไว้ได้ ส่งผลให้การแสดงผลที่มีลักษณะคล้ายคลึงหรือซ้ำๆกันเป็นไปอย่างรวดเร็ว

Document Object Model(DOM)

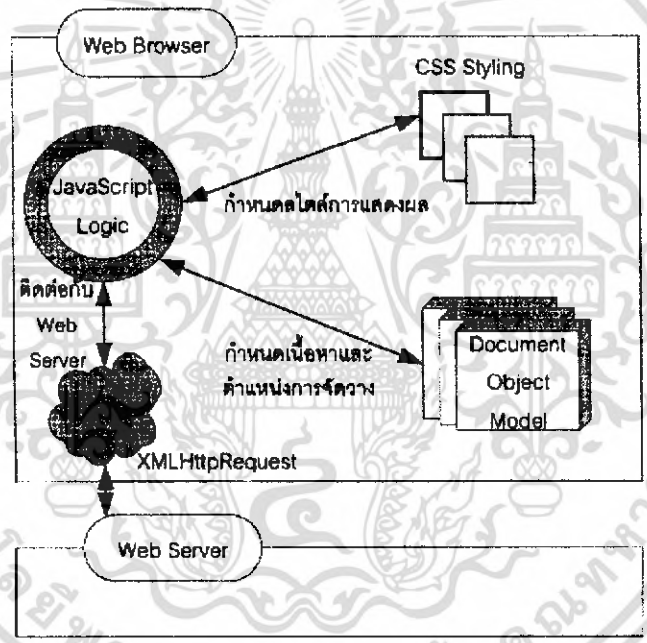
เนื่องจากภาษา JavaScript และสไตล์ชีทไม่สามารถจัดการกับข้อมูลต่างๆในเอกสาร HTML ได้จึงพยายามคิดค้นสิ่งที่จะมาจัดการข้อมูลในเอกสาร HTML จนได้มีการสร้างรูปแบบ DOM ขึ้นมา ซึ่ง DOM สามารถจัดการกับเอกสาร HTML และสามารถลงลึกไปจัดการ Source Code ของภาษา HTML ได้จึงทำให้การสร้างเว็บเพจในปัจจุบันมีความสารามถมากขึ้น Document Object Model(DOM) เป็นแพลตฟอร์มและภาษาสื่อกลางระหว่างโปรแกรมกับสคริปต์ต่างๆสามารถเข้าถึงและปรับปรุง Content โครงสร้างและสไตล์ของเอกสารต่างๆได้ DOM จะมององค์ประกอบของเว็บเพจทั้งหมดเป็นอ็อบเจ็กต์ที่มีการทำงานตามหลัก Object Oriented และ Model ของ DOM จะมอง Element หรือ Tag ต่างๆ เป็น Node ของต้นไม้ โดยโครงสร้างของ DOM จะประกอบไปด้วย อ็อบเจ็กต์และเมธอด จึงทำให้สามารถ

เข้าถึงและปรับปรุงเว็บเพจทุกส่วนได้ และนอกจากนั้น DOM ยังเป็นสคริปต์ที่ประมวลผลบนฝั่งไคลเอนต์(Client Side) เหมือนกับภาษา JavaScript โดยที่ไม่ต้องประมวลผลที่เครื่อง Server จึงทำให้ทำงานได้เร็ว

จากที่กล่าวมาข้างต้นเกี่ยวกับความสามารถของ DOM ทำให้ AJAX สามารถจัดการกับเอกสาร HTML ได้อย่างมีประสิทธิภาพ อีกทั้งทำให้โค้ดของเว็บแอพพลิเคชั่นที่ใช้ AJAX มีความเป็นระเบียบง่ายต่อการจัดการ

XMLHttpRequest Object(XHR Object)

XMLHttpRequest Object ทำหน้าที่ควบคุมการแลกเปลี่ยนข้อมูลระหว่างเว็บเพจ กับเว็บเซิร์ฟเวอร์ โดยข้อมูลที่ใช้แลกเปลี่ยนกันนั้นจะอยู่ในรูปแบบของเอกสาร XML และข้อความสั้นๆ เทคโนโลยีต่างๆที่กล่าวมาถูกนำมาใช้ในเว็บแอปพลิเคชั่นที่สร้างตามแนวคิด AJAX โดยหน้าที่และการทำงานของแต่ละเทคโนโลยีสามารถแสดงให้เห็นอย่างคร่าวๆดังในรูปที่ 2.17



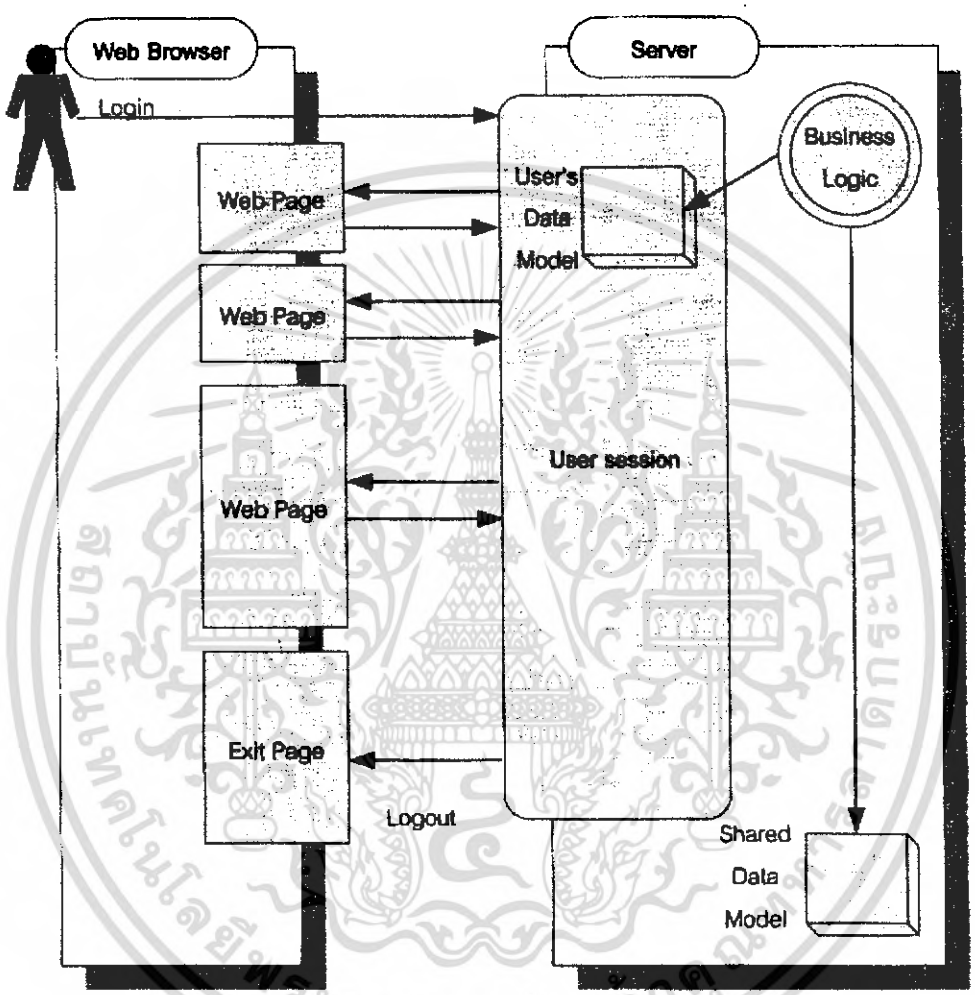
รูปที่ 2.17 แสดงองค์ประกอบของ AJAX

จากรูปที่ 2.17 จะเห็นการทำงานของ AJAX นั้นเทคโนโลยีส่วนใหญ่จะทำงานอยู่ที่ Web Browser โดยมีเทคโนโลยีที่ทำหน้าที่ติดต่อกับเว็บเซิร์ฟเวอร์คือ XMLHttpRequest Object ส่วนการกำหนดเนื้อหาและตำแหน่งการจัดวางเนื้อหาบนเว็บเพจเป็นหน้าที่ของ DOM สำหรับ CSS จะควบคุมสไตร์การแสดงผลของ Web page โดยทั้งหมดจะมี JavaScript เป็นตัวจัดการการแสดงผลข้อมูลต่างๆให้ปรากฏบนเว็บเพจต่อไป

2.2.3 ข้อแตกต่างของเว็บแอปพลิเคชั่นแบบเดิมกับเว็บแอปพลิเคชั่นแบบ AJAX

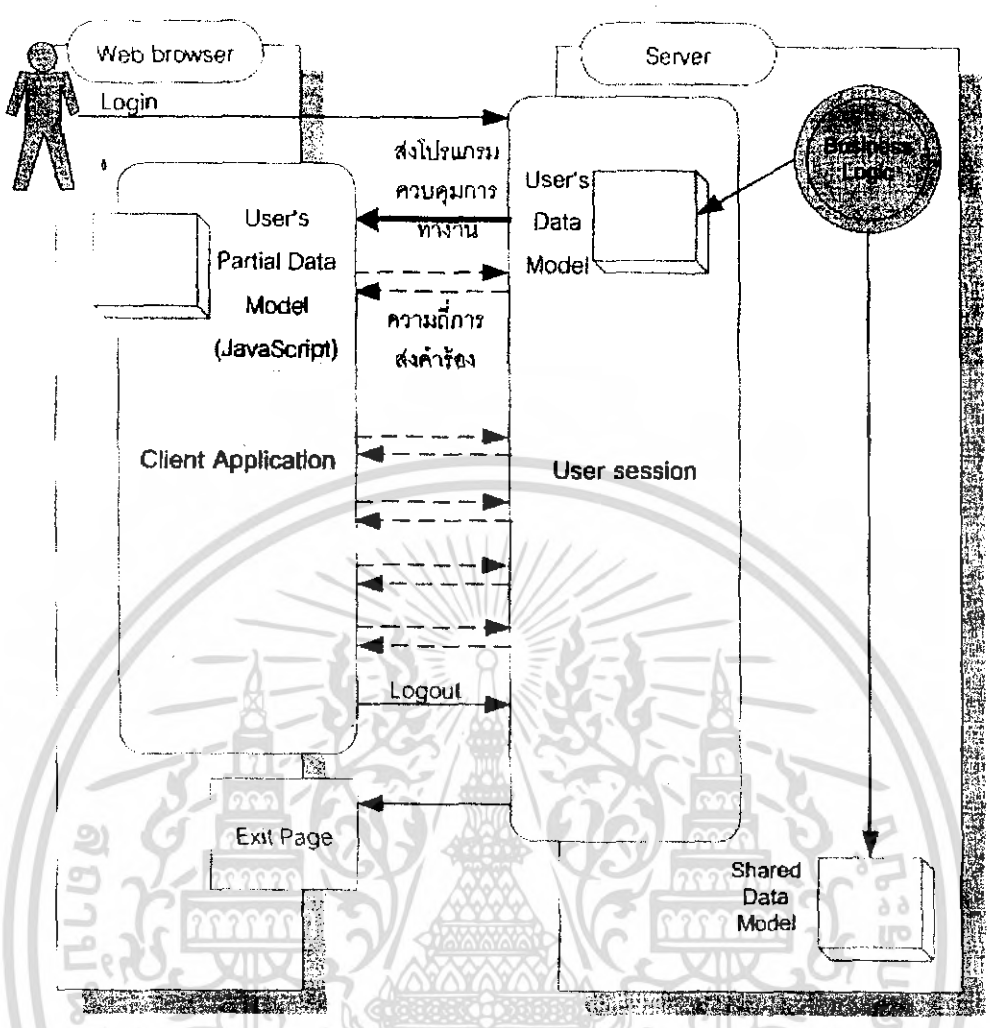
ด้านการทำงานของ Web Browser

สำหรับ Web Browser ตามแบบจำลองการทำงานของเว็บแอปพลิเคชั่นแบบเดิมจะมีหน้าที่แสดงผลของเว็บเพจเท่านั้น Web Browser ไม่สามารถรับรู้หรือมีส่วนเกี่ยวข้องกับกลไกภายในที่จะทำให้ได้ผลลัพธ์นั้นมา เนื่องจากกลไกการตอบสนองต่อ User เกิดขึ้นที่ฝั่ง Server ดังแสดงให้เห็นในรูปที่ 2.18



รูปที่ 2.18 แบบจำลองแสดงการทำงานของ Web Browser สำหรับเว็บแอปพลิเคชั่นแบบเดิม

จากรูปที่ 2.18 เมื่อ User เริ่มล็อกอิน(login)เพื่อร้องขอเว็บเพจจาก Server การทำงานส่วนใหญ่เพื่อให้ได้มาซึ่งการตอบสนองความต้องการของ User จะเกิดขึ้นทางฝั่ง Server จากนั้น User จะได้รับการตอบสนองและได้เว็บเพจกลับมาแสดงที่ Web Browser และเมื่อ User ร้องขอข้อมูลหรือเว็บเพจเพิ่มเติม ก็จะได้รับเว็บเพจขึ้นมาอีกเป็นหน้าใหม่ แต่สำหรับเว็บแอปพลิเคชั่นที่นำ AJAX ไปใช้งานจะย้ายการทำงานบางอย่างออกมาไว้ที่ Web Browser ดังรูปที่ 2.19



รูปที่ 2.19 แบบจำลองการทำงานของ Web Browser สำหรับเว็บแอปพลิเคชั่นแบบ AJAX

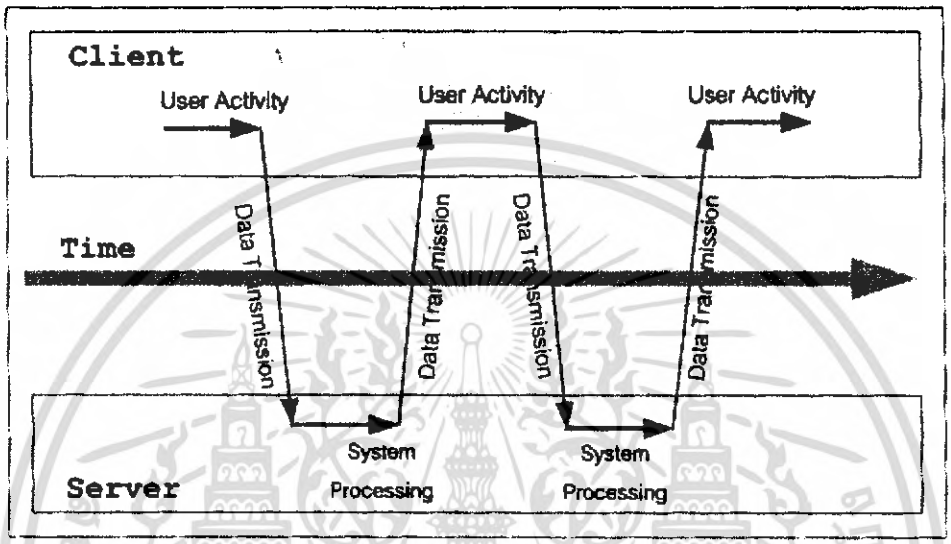
จากรูปที่ 2.19 เมื่อ User เริ่ม Login และร้องขอข้อมูลจาก Server ครั้งแรกนั้น เอกสารที่ส่งมายัง Web Browser จะเป็น JavaScript เอกสารนี้จะยังคงอยู่ที่ Web Browser ตลอดเวลา เพื่อตอบโต้กับ User โดยแบ่งเป็น 2 กรณี คือตอบสนอง User แบบทันทีโดยไม่ส่งคำร้องขอไปยังฝั่ง Server หรือ ส่งคำร้องไปที่ฝั่ง Server (เช่น การเข้าถึงฐานข้อมูลหรือทรัพยากรอื่นๆ)หรือในบางครั้งอาจทำทั้งสองอย่างพร้อมกัน (กรณีที่มีการส่งและรับข้อมูลจาก Server จะมี XMLHttpRequest Object ทำหน้าที่จัดการให้)

ด้านเอกสารที่ใช้แลกเปลี่ยน

การแลกเปลี่ยนข้อมูลระหว่าง Web Browser กับ Server ของเว็บแอปพลิเคชั่นแบบเดิมนั้น ในแต่ละเว็บเพจจะประกอบด้วยข้อมูลจำนวนมาก แต่สำหรับเว็บแอปพลิเคชั่นแบบ AJAX ข้อมูลที่อยู่ในเว็บเพจจะถูกส่งในปริมาณที่มากเฉพาะเว็บเพจแรกเท่านั้น สำหรับข้อมูลที่ส่งตอบกลับครั้งต่อๆไปจะเป็นเอกสารที่มีปริมาณน้อยมาก เนื่องจากเป็นข้อมูลที่ User ต้องการจริงๆซึ่งอยู่ในรูปของเอกสาร XML และข้อความสั้นๆเท่านั้น

ด้านความเร็ว

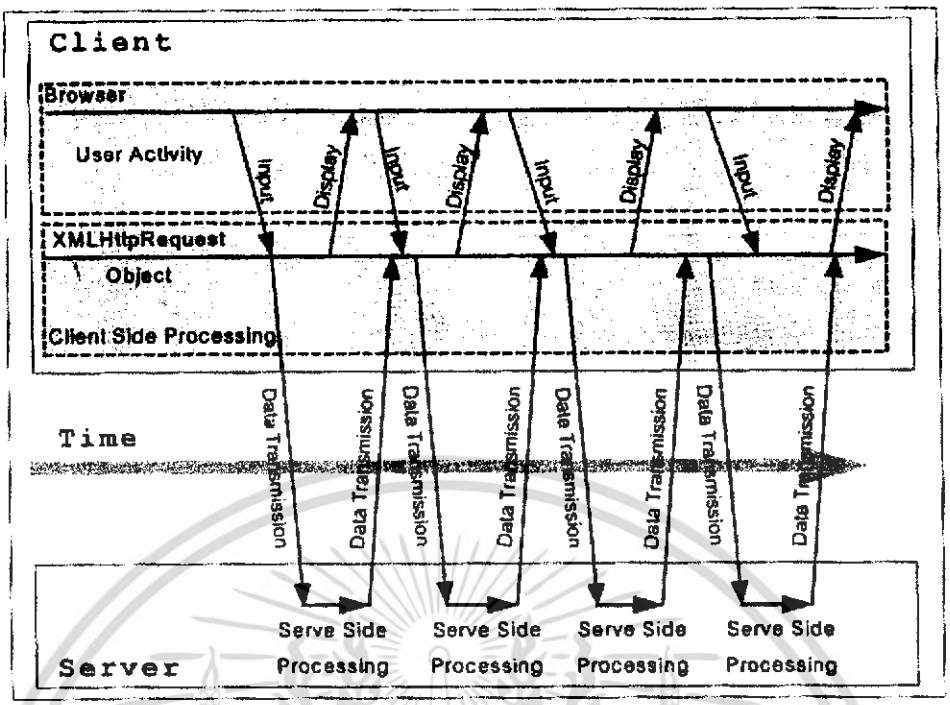
การทำงานเว็บแอปพลิเคชั่นแบบเดิมนั้นทุกครั้งที่ User ร้องขอข้อมูลจาก Server การร้องขอเริ่มที่ User ส่งคำร้องไปยัง Server จากนั้น User ต้องรอจนกว่า Server ทำการประมวลผลข้อมูลเสร็จ และตอบสนองกลับมายัง User เมื่อ User มีการร้องขอข้อมูลใหม่ก็จะต้องรอการตอบกลับเหมือนที่กล่าวมา วิ่งเรียกการทำงานแบบนี้ว่า "Synchronous" และการรอนี้เองทำให้เกิดความล่าช้าในการตอบสนอง User ของการใช้เว็บแอปพลิเคชั่นแบบเดิม ดังรูปที่ 2.20



รูปที่ 2.20 แสดงการทำงานแบบ Synchronous บนเว็บแอปพลิเคชั่นแบบเดิม

จากรูปที่ 2.20 แสดงการทำงานแบบ Synchronous บนเว็บแอปพลิเคชั่นแบบเดิม การทำงานจะเริ่มจาก User เป็นผู้ส่งคำร้อง(Request) ไปยัง Server จากนั้น User ต้องรอการตอบสนองจาก Server ระหว่างที่รอนั้น User จะส่งคำร้องมายัง Server เดิมอีกไม่ได้ User ต้องรอจนกว่าServer ตอบสนองการร้องขอที่ส่งในครั้งแรกให้เสร็จก่อน และเมื่อ Server ตอบสนองการร้องขอดังกล่าวมายัง User แล้ว การสื่อสารระหว่าง User กับ Server จะสิ้นสุดทันที ดังนั้นหาก User ต้องการร้องขอข้อมูลจาก Server เดิมอีก User จะต้องเริ่มส่งคำร้องไปยัง Server แล้วรอการตอบสนองจาก Server ซึ่งเป็นกระบวนการซ้ำๆ แบบนี้ไปเรื่อยๆ

สำหรับการทำงานของ AJAX จะมีเทคโนโลยีที่เรียกว่า "XMLHttpRequest Object" ซึ่งอยู่ฝั่ง Web Browser ทำหน้าที่ช่วยทำงาน คือทุกครั้งที่ User มีการร้องขอข้อมูล XMLHttpRequest Object จะรับการร้องขอและส่งการร้องขอต่อไปยัง Server ในขณะเดียวกัน XMLHttpRequest Object จะค่อยเชื่อมต่อ Server ตลอดเวลา เมื่อ User มีการร้องขอข้อมูลจาก Server จะได้รับการตอบสนองอย่างรวดเร็วโดยไม่ต้องรอ ซึ่งเรียกการทำงานแบบนี้ว่า "Asynchronous"ดังรูปที่ 2.21



รูปที่ 2.21 แสดงการทำงานแบบ Asynchronous บนเว็บแอปพลิเคชั่นแบบ AJAX

รูปที่ 2.21 แสดงการทำงานแบบ Asynchronous บนเว็บแอปพลิเคชั่นแบบ AJAX การทำงานจะเริ่มจาก User เป็นผู้ส่งคำร้อง (Request) โดยใช้ JavaScript ส่งคำร้องไปที่ XMLHttpRequest Object จากนั้น XMLHttpRequest Object จะเลือกว่าจะส่งคำร้องไปยัง Server หรือไม่ หากบางกรณีถ้า XMLHttpRequest Object สามารถตอบสนองคำร้องได้ XMLHttpRequest Object ก็ตอบสนองทันที เช่น การแก้ไขข้อมูลในหน่วยความจำ หรือการแก้ข้อมูลที่ XMLHttpRequest Object มีอยู่แล้ว เป็นต้น หากคำร้องที่ XMLHttpRequest Object รับมานั้นจำเป็นต้องขอข้อมูลจากฐานข้อมูล หรือข้อมูลที่ต้องอาศัยการประมวลผลจาก Server เป็นต้น และเมื่อ Server ตอบสนองไปยัง XMLHttpRequest Object แล้ว XMLHttpRequest Object จะส่งต่อการตอบสนองไปยัง User ด้วย JavaScript ถึงแม้ว่า XMLHttpRequest Object จะได้รับการตอบสนองจาก Server แล้วแต่การติดต่อสื่อสารระหว่าง XMLHttpRequest Object และ Server ยังคำดำเนินต่อไปเพื่อการส่งคำร้องที่ต้องขอข้อมูลจาก Server ต่อไป จนกว่า User จะหยุดการร้องขอและปิดเว็บเพจไป

ด้านการใช้งาน JavaScript

ถึงแม้ว่าในเว็บแอปพลิเคชั่นแบบเดิม จะมีการนำ JavaScript ไปใช้งานบ้าง โดยนำไปใช้เพิ่มลูกเล่นให้กับเว็บเพจหรือทำการโต้ตอบบางอย่างกับ User เช่น การแจ้งข้อความเตือนเมื่อ User ป้อนข้อมูลไม่ครบ เป็นต้น ซึ่งการทำงานของ JavaScript ที่อยู่ในเว็บแอปพลิเคชั่นแบบเดิม จะถูกเรียกมาใช้งานบางเวลาหรือบางช่วงเท่านั้น ขึ้นอยู่กับกิจกรรมที่ User กระทำ แต่ในเว็บแอปพลิเคชั่นที่นำ AJAX ไปใช้งานนั้น JavaScript จะทำงานอยู่ตลอดเวลา ความเร็วในการทำงานเป็นไปอย่างต่อเนื่อง

และใช้หน่วยความจำ(Memory) อยู่ตลอดเวลาเช่นกัน จะเห็นได้ว่าการใช้งาน JavaScript ใน AJAX นั้นค่อนข้างหนักกว่าการใช้งานในเว็บแอปพลิเคชั่นแบบเดิม ด้วยสาเหตุนี้การเขียนโค้ดสำหรับ AJAX จึงจำเป็นต้องอาศัยเว็บโปรแกรมเมอร์ที่มีความชำนาญเป็นอย่างมาก เพื่อให้สามารถเขียนโค้ดได้อย่างเป็นระเบียบ สามารถปรับปรุงแก้ไขได้ง่ายในภายหลัง ทำให้เป็นเว็บแอปพลิเคชั่นที่มีประสิทธิภาพ

2.2.4 ทักษะที่จำเป็นต่อการใช้งาน AJAX

จากที่กล่าวมาแล้วว่า AJAX เป็นการนำเทคโนโลยีต่างๆมาใช้งานร่วมกัน ดังนั้นผู้พัฒนาเว็บแอปพลิเคชั่น ที่จะนำ AJAX ไปใช้งานจำเป็นต้องมีความรู้หรือทักษะพื้นฐานดังต่อไปนี้

1. ทักษะด้านการเขียนโค้ดผู้พัฒนาต้องสามารถเขียนโค้ด DHTML เองได้โดยไม่ใช้เครื่องมือ นอกจากนี้เว็บโปรแกรมเมอร์ยังต้องมีความรู้ทางด้าน XHTML พอสมควรอีกด้วย
2. ทักษะการใช้ CSS เว็บโปรแกรมเมอร์จะต้องสามารถจัด Layout ของเว็บเพจด้วย CSS ได้เป็นอย่างดีเนื่องจากช่วยให้โค้ดเป็นระเบียบเรียบร้อยง่ายต่อการนำไปพัฒนาต่อ
3. ทักษะการใช้ JavaScript เว็บโปรแกรมเมอร์ต้องมีความชำนาญในการเขียน JavaScript เป็นอย่างมากต้องเข้าใจหลักการของโครงสร้างข้อมูลของ JavaScript ได้เป็นอย่างดี
4. ทักษะการใช้งาน DOM เว็บโปรแกรมเมอร์ จะต้องมีความรู้ความเข้าใจเกี่ยวกับ DOM อย่างถ่องแท้ ซึ่งรวมถึง XPATH และ XSLT
5. ทักษะเกี่ยวกับ XMLHttpRequest Object รวมทั้งความแตกต่างของ Web Browser ในแต่ละรุ่นว่ามีความแตกต่างกันอย่างไร ซึ่งจะเป็นประโยชน์ต่อการนำ AJAX ไปใช้งานเป็นอย่างยิ่ง

สำหรับผู้ใช้ที่มีความรู้ความเข้าใจเกี่ยวกับเทคโนโลยีต่างๆตามที่กล่าวมาแล้ว สามารถนำ AJAX ไปช่วยพัฒนาเว็บแอปพลิเคชั่นได้ตามความต้องกา

2.3 การใช้งาน XMLHttpRequest Object

การทำงานของ AJAX เป็นการทำงานร่วมกันของหลายๆเทคโนโลยี ได้แก่ DHTML,CSS,JavaScript,DOM,XML และ XMLHttpRequest Object ถือว่าเป็นเทคโนโลยีที่สำคัญของ AJAX ซึ่งทำหน้าที่ควบคุมการแลกเปลี่ยนข้อมูลระหว่าง Web Browser กับฝั่ง Server เนื้อหาในส่วนนี้จะกล่าวแนะนำและอธิบายเกี่ยวกับหลักการทำงานของ XMLHttpRequest Object โดยแบ่งเป็นหัวข้อดังนี้

2.3.1 แนะนำ XMLHttpRequest Object

XMLHttpRequest Object เป็นเทคโนโลยีที่ทำหน้าที่การแลกเปลี่ยนข้อมูลระหว่าง Web Browser กับเว็บเซิร์ฟเวอร์(Web Server) โดยที่เอกสารที่ใช้แลกเปลี่ยนกันนั้นจะอยู่ในรูปเอกสาร XML ตามที่กล่าวมาแล้วว่า AJAX เป็นเพียงเทคนิคการนำหลายๆเทคโนโลยีมาทำงานร่วมกัน ซึ่งเทคโนโลยีที่ AJAX นำมาใช้งานนั้นส่วนใหญ่ใช้งานแอพพลิเคชั่นแบบเดิมอยู่แล้ว มีเพียง XMLHttpRequest Object

เท่านั้นที่เป็นเทคโนโลยีใหม่ โดย AJAX ใช้ JavaScript เขียนโค้ดเพื่อควบคุมการทำงานของ XMLHttpRequest Object เนื่องจากการใช้งาน XMLHttpRequest Object กับ Web Browser แต่ละรุ่นมีการใช้งานที่ต่างกัน คือ Internet Explorer นำ XMLHttpRequest Object ไปใช้กับส่วนที่เรียกว่า "ActiveX Object" แต่หากเป็น Web Browser รุ่นอื่น เช่น Firefox,Safari และ Opera จะนำ XMLHttpRequest Object ไปใช้งานกับส่วนที่เรียกว่า "Native JavaScript Object"ดังนั้นสิ่งแรกที่ต้องทำในขั้นตอนการเขียนโค้ด คือ การกำหนดฟังก์ชัน(Function) เพื่อเลือกใช้ XMLHttpRequest Object ตามขั้นตอนต่อไปนี้

ขั้นตอนแรกในการเรียกใช้ XMLHttpRequest Object คือ ต้องเขียนโค้ดตรวจสอบการทำงานของ Web Browser ที่ User ใช้ เพื่อให้ทราบว่าสามารถสนับสนุน ActiveX Object หรือไม่หาก Web Browser ที่ User ใช้นั้นสนับสนุน ActiveX Object แสดงว่าสามารถเรียกใช้ XMLHttpRequest Object กับ ActiveX Object ได้ แต่หาก Web Browser ที่ User ใช้นั้นไม่สนับสนุน ActiveX Object ก็สามารถใช้ XMLHttpRequest Object กับ Native JavaScript Object ดังตัวอย่างที่ 2.1

ตัวอย่างที่ 2.1 แสดงโค้ดที่ใช้กำหนดเงื่อนไขการใช้งาน XMLHttpRequest Object

```
1.  var xmlHttp;
2.
3.  function createXMLHttpRequest Object(){
4.      if(windows.ActiveXObject){      :เงื่อนไขตรวจสอบการสนับสนุน ActiveX Object
5.        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
6.        }
7.        else if (windows.XMLHttpRequest){ :เงื่อนไขเมื่อตรวจสอบไม่พบการสนับสนุน Active Object
8.        xmlHttp = new XMLHttpRequest();
9.        }
10. }
```

การเขียนโค้ดตามที่แสดงในตัวอย่างที่ 2.1 จะเห็นได้ว่ามีการทำงานที่สำคัญ 2 ส่วน คือ ส่วนแรกเป็นการกำหนดเงื่อนไขการเรียกใช้ XMLHttpRequest Object สำหรับ Web Browser ที่สนับสนุน ActiveX Object ตามคำสั่งที่ปรากฏในบรรทัดที่ 4-5 ถ้าเงื่อนไขตรวจสอบแล้วปรากฏว่า Web Browser ที่ User ใช้นั้นไม่สนับสนุน ActiveX Object แล้วการเรียกใช้ XMLHttpRequest Object จะเป็นไปตามเงื่อนไขในส่วนที่สองตามที่ปรากฏในบรรทัดที่ 7-8 ซึ่งเป็นการเรียกใช้ XMLHttpRequest Object สำหรับ Web Browser ที่สนับสนุน Native JavaScript Object

Methods และ Properties ของ XMLHttpRequest Object

ในการใช้งาน XMLHttpRequest Object สามารถเรียกใช้ Methods ได้หลาย Methods ดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 แสดง Methods ของ XMLHttpRequest Object

ชื่อ Methods	คำอธิบาย
Abort()	หยุดความต้องการปัจจุบัน
getAllResponseHeader()	รายงานค่าของ Header ทั้งหมดที่ตอบกลับมาสำหรับ

	HTTP Request
getResponseHeader("Header")	รายงานค่าของ Header เฉพาะที่กำหนด
Open("methods", "url",Asynch)	กำหนดการติดต่อ Server โดยมีค่าข้อมูลที่เกี่ยวข้อง 3 ค่า ดังนี้ - ค่า "Method" เป็นการเรียกใช้ Method เช่น GET หรือ POST - ค่า "URL" เป็นที่อยู่ปลายทางที่ที่ต้องการร้องขอ อาจเป็นชื่อของไฟล์เอกสารหรือเป็นชื่อ เว็บเพจก็ได้ - ค่า Asynch เป็นค่ากำหนดการทำงานแบบ Asynchronous ซึ่งเป็นได้ทั้งค่า True และ False
Send(Content)	ส่งคำร้องไปยัง Server
setRequestHeader("Header", "Value")	กำหนด Header และค่าของ Header ใน HTTP Request

จากตารางที่ 2.1 แสดงให้เห็นถึง Methods ที่ใช้กับ XMLHttpRequest Object สำหรับการเรียกใช้งานแต่ละ Methods นั้นขึ้นอยู่กับความต้องการของเว็บโปรแกรมเมอร์ และในส่วนนี้มีตัวอย่างการเรียกใช้ Methods ในการส่ง Request ด้วย XMLHttpRequest Object ดังการในหัวข้อถัดไป

การใช้งาน XMLHttpRequest Object นอกจากจะทราบ Methods ตามที่กล่าวมาแล้ว เว็บโปรแกรมเมอร์ยังต้องทราบเกี่ยวกับ Properties ที่ใช้ด้วย ดังแสดงในตารางที่ 2.2

ตารางที่2.2 แสดง Properties ที่ใช้กับ XMLHttpRequest Object

ชื่อ Properties	คำอธิบาย
onreadystatechange	ตัวควบคุมเหตุการที่เกิดการเปลี่ยนแปลงของสถานการณ์ทำงาน
readyState	ตัวเลขแสดงสถานการณ์ทำงาน 0 หมายถึง ยังไม่เริ่มทำงาน 1 หมายถึง กำลังโอนถ่ายข้อมูล 2 หมายถึง โอนถ่ายข้อมูลเสร็จแล้ว 3 หมายถึง อยู่ระหว่างการทำงาน 4 หมายถึง เสร็จสิ้นการทำงานโดยสมบูรณ์
responseText	ข้อมูลที่ส่งจาก Server ที่เป็นข้อความ

responseXML	ข้อมูลที่ส่งมาจาก Server ที่เอกสาร XML และสามารถจัดการได้ด้วย DOM
status	รหัสตัวเลขที่ส่งมาจาก Server เช่น "200" หมายถึง ตกลง และ "400"หมายถึง ค้นหาไม่พบ
statusText	รหัสข้อความที่ส่งมาจาก Server เช่น "OK" หรือ "Not Found"

2.3.2 วิธีใช้งาน XMLHttpRequest Object

วิธีใช้งาน XMLHttpRequest Object ที่กล่าวต่อไปนี้เป็นตัวอย่างการส่งคำร้อง (Request) จาก Web Browser ไปยังฝั่ง Server โดย Request ขอไฟล์เอกสาร XML ที่มีขนาดของไฟล์ไม่ใหญ่นัก ดังนี้

ตัวอย่างการส่ง Request ด้วย XMLHttpRequest Object

ตัวอย่างต่อไปนี้เป็นการสร้างเอกสาร HTML ที่มีปุ่มโต้ตอบกับ User เพียง 1 ปุ่ม เมื่อ User คลิกเมาส์ที่ปุ่มดังกล่าวการเชื่อมต่อระหว่าง Web Browser กับฝั่ง Server จะเกิดขึ้นทันที เมื่อฝั่ง Server ได้รับ Request และตอบสนองกลับมายัง Web Browser ด้วยข้อมูลที่อยู่ในรูปของ XML ที่แสดงข้อความสั้นๆบนหน้าต่างแสดงข้อความ ดังตัวอย่างที่ 2.2

ตัวอย่างที่ 2.2 แสดงโค้ดการส่ง Request ของไฟล์ SimpleRequest.html

```
1.  <!DOCTYPE html PUBIC "-//W3C//DTD XHTML 1.0Strict/EN"
2.  http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>
3.  <html xhtml= http://www.w3.org/1999/xhtml>
4.  <head>
5.  <title>ตัวอย่างการใช้ XMLHttpRequest Object</title>
6.  <script type= "text/javascript">
7.  var xmlHttp;
8.  function createXMLHttpRequest(){
9.     if (window.ActiveXObject){
10.       xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
11. }
12. else if (window.XMLHttpRequest){
13.       xmlHttp = new XMLHttpRequest();
14.    }
15. }
16. function startRequest(){
17. createXMLHttpRequest();
18.       xmlHttp.onreadystatechange = handleStateChange;
19.       xmlHttp.open("GET","Response.xml",true);
20.       xmlHttp.send(null);
21. }
22.
23. function handleStateChange(){
24.    if(xmlHttp.readyState == 4){
25.       if(xmlHttp.status == 200){
26.         alert("ข้อความตอบสนองจาก Server:" +xmlHttp.responseText);
27.         }
28.      }
29. }
30. </script>
31. </head>
```

```
32. <body>
33. <from action= "#">
34.     <input type= "button" value= "คลิกเพื่อส่ง Request" onclick= "startRequest();"/>
35. </from>
36. </body>
37. </html>
```

จากตัวอย่างที่ 2.2 แสดงโค้ดของไฟล์ SimpleRequest.html ที่เป็นตัวอย่างของการส่งคำร้อง (Request) จาก Web Browser ไปยัง Web Server โดย Request ที่ส่งไปนั้นถูกสร้างขึ้นมาอย่างง่ายๆ ไม่ต้องสร้าง Form สำหรับป้อนข้อมูลเพียงแค่ส่ง Request ของเอกสาร XML ที่อยู่ฝั่ง Server เท่านั้น สำหรับขั้นตอนการทำงานสรุปได้ดังนี้

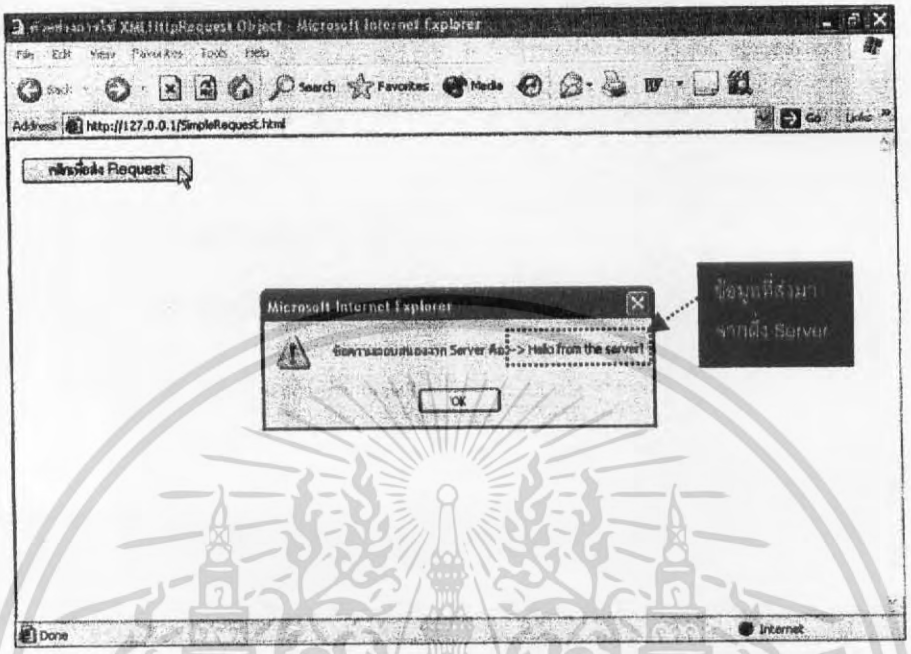
1. กำหนดฟังก์ชั่นชื่อ "createXMLHttpRequest()" เพื่อตรวจสอบ Web Browser ที่ User ใช้ โดยการกำหนดเงื่อนไขการเรียกใช้ XMLHttpRequest Object สำหรับ Web Browser ที่สนับสนุน ActiveX Object (ดังโค้ดในบรรทัดที่ 9-10) ถ้าเงื่อนไขตรวจสอบเสร็จแล้วปรากฏว่า Web Browser ที่ User ใช้นั้นไม่สนับสนุน ActiveX Object แล้วจะเรียกใช้ XMLHttpRequest Object สำหรับ Web Browser ที่สนับสนุน Native JavaScript Object(ดังคำสั่งในบรรทัดที่ 12-13)
2. กำหนดฟังก์ชั่นที่จะใช้ควบคุมการเปลี่ยนแปลงของสถานการณ์ทำงาน โดยกำหนด Properties ชื่อ "onreadystaechange"(ดังโค้ดในบรรทัดที่ 16-18)
3. กำหนด Method ของ XMLHttpRequest Object เป็น "open()" โดยมี Parameter ที่จะส่งไปยัง Server ด้วยกัน 3 ค่า ได้แก่ ค่าแรกเป็นข้อความแสดงรูปแบบการร้องขอ(GET หรือ POST) ในที่นี้ใช้ GET (สำหรับหารเลือกค่าใช้ GET หรือ POST มีอธิบายในหัวข้อถัดไป) ค่าที่สองเป็นข้อความแสดงปลายทางที่ต้องการร้องขอ URL ในที่นี้เป็นเอกสารชื่อ "Response.xml" และค่าที่สามเป็นค่าบูลีน(Boolean) ซึ่งเป็นค่าคงที่เชิงตรรกะมี 2 ค่า คือ จริง(True)และเท็จ(False) เพื่อใช้กำหนดการทำงานแบบ Asynchronous (หากกำหนดเป็น จริง แสดงว่าสามารถร้องขอติดต่อกับฝั่ง Server ได้โดยไม่ต้องรอการตอบกลับจาก Server ซึ่งก็คือการทำงานแบบ Asynchronous นั่นเอง แต่หากกำหนดเป็น เท็จ แสดงว่าการร้องขอครั้งต่อไปจะเกิดขึ้นได้ก็ต่อเมื่อได้รับการตอบสนองจาก Server แล้วเท่านั้น ซึ่งก็คือการทำงานแบบ Synchronous นั่นเอง)ในที่นี้ใช้ค่า จริง ดังโค้ดในบรรทัดที่ 19
4. ส่ง Request ไปยังฝั่ง Server โดยใช้ Method ที่ชื่อ "Send()" ซึ่ง Parameter ที่ใช้ใน Method นี้มีได้เพียง 1 Parameter เท่านั้น อาจเป็นข้อความหรือ DOM Object ก็ได้ ค่า Parameter ดังกล่าวจะถูกส่งไปยัง URL ปลายทาง(เป็นส่วนประกอบใน Request) ในที่นี้ใช้ Parameter เป็น "null"

เมื่อเขียนโค้ดควบคุมการทำงานตามที่กล่าวมาแล้ว ขั้นตอนต่อไปต้องเตรียมเอกสาร XML ไว้ที่ฝั่ง Server เพื่อตอบสนองกลับไปยัง Web Browser ในที่นี้ใช้เอกสาร XML ที่ชื่อ "Response.xml"

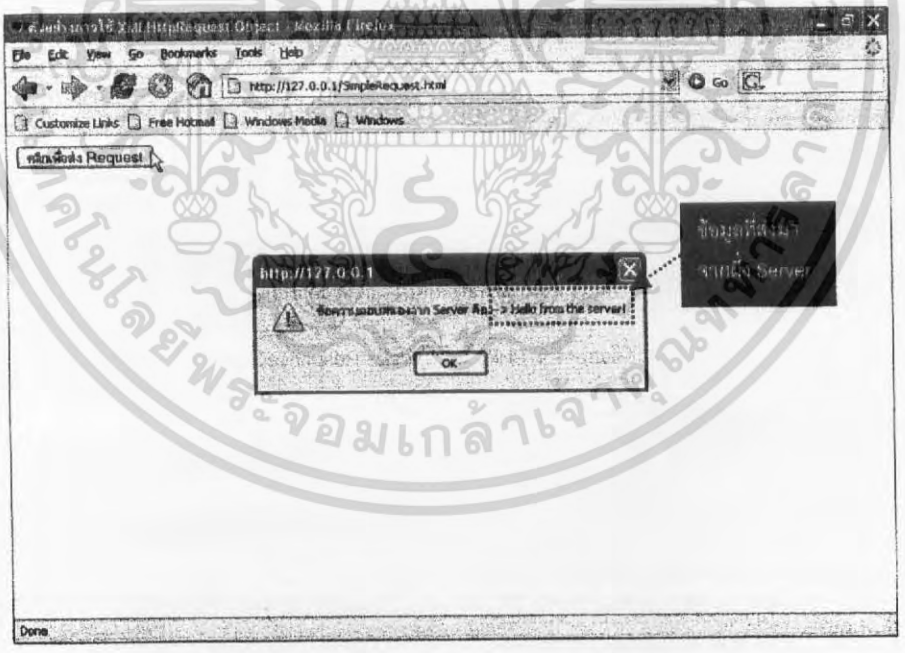
ตัวอย่างที่ **2.3** แสดงโค้ดของไฟล์ **Response.xml**

```
1. Hello from the server!
```

จากตัวอย่างที่ 2.3 แสดงโค้ดของไฟล์ Response.xml ซึ่งข้อความในเอกสารคือ "Hello from the server!" เราสามารถเห็นหน้าจอการส่ง Request เปรียบเทียบกันระหว่าง Web Browser 2 รุ่นคือ Internet Explorer และ Mozilla Firefox ดังรูปที่ 2.22 และ 2.23



รูปที่ 2.22 แสดงผลการส่ง Request บน Internet Explorer



รูปที่ 2.23 แสดงผลการส่ง Request บน Firefox

จากการแสดงผลในรูปที่ 2.22 และ 2.23 แสดงให้เห็นผลลัพธ์ที่เกิดจากการทำงานของไฟล์ SimpleRequest.html ที่สามารถใช้ XMLHttpRequest Object ในการส่ง Request ซึ่งผลลัพธ์ดังกล่าวคือ ข้อความ "Hello from the server!" โดยสามารถแสดงผลลัพธ์ทั้งบน Internet Explorer และ Firefox ได้อย่างไม่มีปัญหา โดยมีผลมาจากการกำหนดฟังชันด้วย JavaScript เพื่อตรวจสอบการทำงานของ Web Browser ทำให้สามารถเรียกใช้ XMLHttpRequest Object ได้อย่างเหมาะสมกับ Web Browser ที่แตกต่างกัน

2.3.3 การส่ง Request แบบ GET และ POST

สำหรับการส่ง Request จาก Web Browser ไปยังฝั่ง Server ในเว็บแอปพลิเคชั่นแบบ AJAX นั้นจะมี XMLHttpRequest Object ทำหน้าที่สร้างและส่ง Request โดยสามารถทำได้ 2 วิธีคือ GET และ POST

การส่ง Request แบบ GET เป็นการส่งข้อมูลผ่านทาง URL ของเว็บเพจ เช่น http://localhost/test/login.php?username=kafaak&password=123456 เป็นต้น หรือเป็นการส่ง Request เพื่อขอข้อมูลโดยระบุ URL ปลายทางโดยตรง

การส่ง Request แบบ POST เป็นการส่งข้อมูลจากฟอร์มไปยังสคริปต์โดยตรง โดยไม่ผ่าน URL และมีการเข้ารหัสข้อมูลก่อนเพื่อป้องกันข้อมูลสูญหายระหว่างการส่ง

ตัวอย่างการส่ง Request แบบ GET

ตัวอย่างการส่ง Request ต่อไปนี้เป็นการใช้งาน AJAX ร่วมกับภาษา PHP (เมื่อนำไปใช้งานจริงอาจใช้ภาษาอื่นก็ได้)ซึ่ง AJAX ใช้ XMLHttpRequest Object ช่วยส่ง Request จาก Web Browser ไปยังฝั่ง Server ซึ่งมีไฟล์ของ PHP ทำหน้าที่รับข้อมูลจาก Request ดังกล่าวและตอบสนองข้อมูลกลับมายัง Web Browser ตามขั้นตอนต่อไปนี้

ขั้นตอนแรกผู้ใช้ต้องสร้างไฟล์ด้วยภาษา PHP ในที่นี้ใช้ชื่อ "ResponseGet.php"ซึ่งการทำงานของไฟล์ PHP จะทำงานที่ฝั่ง Server

ตัวอย่างที่ 2.4 แสดงโค้ดของไฟล์ ResponseGet.php

```
1. <?php
2. header("content-Type:text/plain;charset=TIS-620");
3. //รับข้อมูลมาเก็บในตัวแปร $dataGet
4. $dataGet = $_GET["dataget"];
5. //แสดงค่าข้อมูลที่รับมา
6. echo "Server ได้รับ $dataGet แล้ว";
7. ?>
```

จากตัวอย่าง 2.4 แสดงโค้ดของไฟล์ ResponseGet.php ซึ่งเขียนด้วยภาษา PHP (ทำหน้าที่ฝั่ง Server)มีหน้าที่รับ Request ที่ส่งมาจาก XMLHttpRequest Object ซึ่งโค้ดเขียนด้วยภาษา PHP นั้นเป็นโค้ดสั้นๆ ไม่มีโค้ดของ HTML และ JavaScript รวมอยู่ เนื่องจากโค้ดที่เป็น HTML และ JavaScript

ส่วนใหญ่จะถูกใช้ที่ไฟล์ HTML จึงทำให้สามารถจัดการกับโค้ดของไฟล์ ResponseGet.php ได้ง่ายและแสดงขั้นตอนการทำงานของโค้ดได้อย่างชัดเจน

ขั้นตอนต่อไป คือ การสร้างเว็บเพจ เพื่อให้ User ป้อนข้อมูล ในที่นี้เป็นไฟล์ HTML ชื่อ "XMLHttpGetRequest.html"ที่ใช้ JavaScript ควบคุมการทำงาน และโค้ดที่ต้องเขียนก่อนจะเป็นโค้ดเกี่ยวกับการเลือกใช้ XMLHttpRequest Object ให้เหมาะกับรุ่นของ Web Browser ที่ User กำลังใช้งาน (ดังที่กล่าวมาแล้วในตัวอย่างที่ 2.1) จากนั้นจึงเขียนโค้ดที่ควบคุมการทำงานต่างๆ

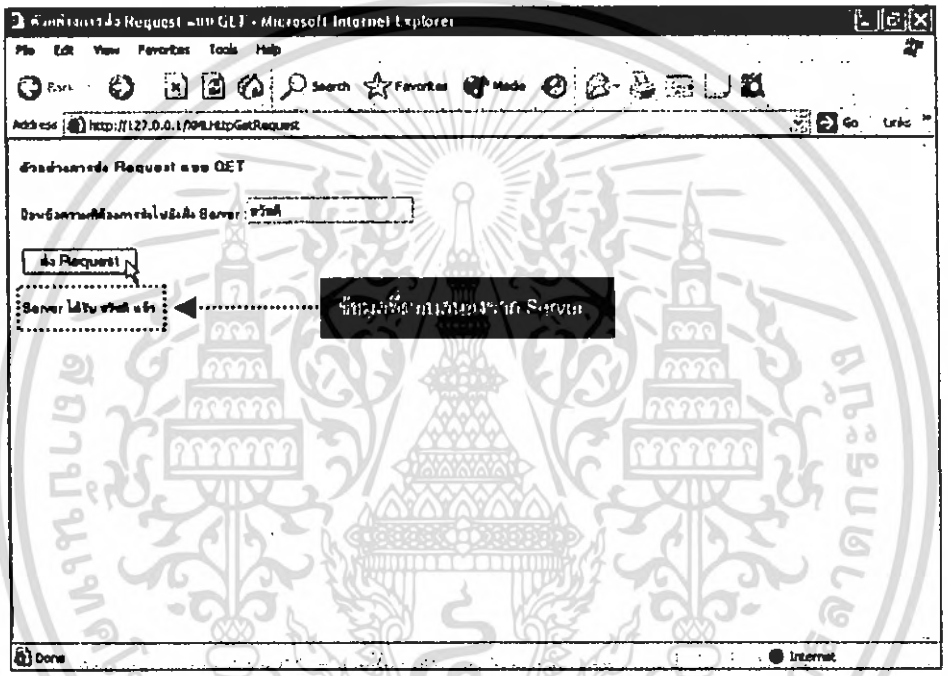
ตัวอย่างที่2.5 แสดงโค้ดของไฟล์ XMLHttpGetRequest.html

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2.  <html>
3.  <head>
4.  <meta http-equiv= 'content-type' content= 'text/html' charset= 'windows-874'>
5.  <title>ตัวอย่างการส่ง Request แบบ GET </title>
6.
7.  <script type = "text/javascript">
8.      var xmlHttp;
9.
10. function createXMLHttpRequest(){
11.     if(window.ActiveXObject){
12.             xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
13.     }
14.     else if (window.XMLHttpRequest){
15.             xmlHttp = new XMLHttpRequest();
16.     }
17. }
18.
19. function requestCustomerinfo(){
20.     createXMLHttpRequest();
21.     var sText = document.getElementById("datage").value;
22.     xmlHttp.open("get", "ResponseGet.php?dataget = "+sText,true);
23.     xmlHttp.onreadystatechange = function(){
24.       if(xmlHttp.readyState == 4){
25.         if(xmlHttp.status == 200){
26.             displayinfo(xmlHttp.responseText);
27.         }else{
28.             displayinfo("พบข้อผิดพลาด:"+xmlHttp.statusText);
29.         }
30.     }
31. };
32.     xmlHttp.send(null);
33. }
34.
35.     function displayinfo(){
36.         document.getElementById( "divinfo").innerHTML = xmlHttp.responseText;
37.     }
38. </script>
39. </heed>
40. <body>
41. <p><b>ตัวอย่างการส่ง Request แบบ GET </b></p>
42. <p>ป้อนข้อความที่ต้องการส่งไปยังฝั่ง Server:<input type = "text" id= "dataget" value= ""/>
    </p>
43. <p><input type = "button" value = "ส่ง Request" onClick= "startRequest()"/></p>
44. <div id= "divinfo"></div>
45. </body>
46. </html>
```

จากตัวอย่างที่ 2.5 ที่แสดงโค้ดของไฟล์ XMLHttpGetRequest.html โดยแสดงให้เห็นถึงรูปแบบการส่ง Request แบบ GET ซึ่งระบุค่าข้อมูลที่ User ต้องการผ่านทาง URL ด้วยโค้ดแสดงในบรรทัดที่ 22 ดังนี้

```
OXmlHttp.open( "get", "ResponseGet.php?dataget= "+sText.true);
```

ด้วยโค้ดที่ส่ง Request แบบ GET ดังกล่าว เมื่อ User ป้อนข้อมูลที่ต้องการผ่าน Web Browser จะได้รับการตอบสนองจาก Server ซึ่งผลลัพธ์ดังกล่าวจะถูกนำมาเก็บไว้ในแท็กชื่อ "div" โดยในที่นี้กำหนดตัวแปรชื่อ "divinfo" เพื่อเก็บผลลัพธ์ดังโค้ดในบรรทัดที่ 44 (สามารถเปลี่ยนชื่อตัวแปรเป็นชื่ออื่นได้) จากนั้นข้อมูลในตัวแปรดังกล่าวจะถูกนำไปแสดงผล(ดังโค้ดในบรรทัดที่ 36)แสดงผลลัพธ์ได้ดังรูปที่ 2.24



รูปที่ 2.24 แสดงเว็บเพจ ตอบสนอง User ของไฟล์ XMLHttpGetRequest.html

จากรูปที่ 2.24 แสดงการตอบได้ User ของไฟล์ XMLHttpGetRequest.html เมื่อ User ป้อนข้อมูลแล้วคลิกที่ปุ่ม "ส่ง Request" การส่ง Request แบบ GET จะเริ่มทำงานและตอบสนองกลับมายัง Web Browser ให้ User เห็นผลการตอบสนองบนหน้าจอ

ตัวอย่างการส่ง Request แบบ POST

จากตัวอย่างการใช้ XMLHttpRequest Object ส่ง Request แบบ GET ที่กล่าวมาเป็นการส่ง Request ผ่าน URL แต่การใช้ XMLHttpRequest Object ส่ง Request แบบ POST ที่กล่าวต่อไปนี้มีการใช้งานที่แตกต่างกัน คือ การส่ง Request แบบ POST จะใช้ Form เพื่อรับข้อมูลจาก User จากนั้น XMLHttpRequest Object จะนำข้อมูลดังกล่าวส่งไปยังฝั่ง Server ตามขั้นตอนต่อไปนี้

ขั้นแรกต้องสร้างไฟล์ที่ทำหน้าที่รับข้อมูลจาก XMLHttpRequest Object ในที่นี้ใช้ภาษา PHP สร้างไฟล์ชื่อ "ResponsePost.php"เก็บไว้ที่ฝั่ง Server

ตัวอย่างที่ 2.6 แสดงโค้ดของไฟล์ ResponsePost.php

```
1. <?php
2. header("Content-Type:text/plain");
3.
4. //รับข้อมูลมาเก็บในตัวแปร$dataPost
5. $dataPost = $_POST["datapost"];
6.
7. //แสดงค่าข้อมูลที่รับมา
8. echo $dataPost;
9. ?>
```

จากตัวอย่างที่ 2.6 เป็นโค้ดที่เขียนด้วยภาษา PHP สำหรับรับข้อมูลจาก XMLHttpRequest Object ซึ่งข้อมูลที่มาจาก XMLHttpRequest Object จะถูกควบคุมด้วย JavaScript ที่อยู่ในไฟล์ HTML ที่ต้องสร้างในขั้นตอนต่อไป

ขั้นตอนนี้เป็นการสร้างไฟล์ HTML เพื่อใช้เป็นเว็บเพจสำหรับโต้ตอบกับ User ในที่นี้เป็นไฟล์ชื่อ "XMLHttpPostRequest.html" ซึ่งใช้ JavaScript เขียนโค้ดควบคุมการทำงานต่างๆ

ตัวอย่างที่ 2.7 แสดงโค้ดของไฟล์ XMLHttpPostRequest.html

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2.  <html>
3.  <head>
4.  <meta http-equiv= 'content-type' content= 'text/html' charset= 'windows-874'>
5.  <title>ตัวอย่างการส่ง Request แบบ POST</title>
6.   <script type = "text/javascript">
7.      var xmlHttp;
8.
9.  function createXMLHttpRequest(){
10.     if(window.ActiveXObject){
11.          xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
12.     }
13.      else if (window.XMLHttpRequest){
14.          xmlHttp = new XMLHttpRequest();
15.     }
16. }
17.
18. function startRequest(){
19.     createXMLHttpRequest();
20.     var pForm = document.forms[0];
21.     var pBody = getRequestBody(pForm);
22.     xmlHttp.open("post", pForm.action,true);
23.     xmlHttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
24.
25.     xmlHttp.onreadystatechange = function(){
26.       if(xmlHttp.readyState == 4){
27.          if(xmlHttp.status == 200){
28.              saveResult(xmlHttp.responseText);
29.       }else{
30.              saveResult("พบข้อผิดพลาด:"+xmlHttp.statusText);
31.       }
32.     }
33. };
34.     xmlHttp.send(pBody);
```

```
35. }
36.
37.    function getRequestBody(pForm){
38.        var nParams = new Array();
39.
40.        for(var i=0;i<pForm.elements.length;i++){
41.            var pParam = encodeURLComponent(pForm.elements[i].name);
42.            pParam += "=";
43.            pParam += encodeURLComponent(pForm.elements[i].value);
44.            nParams.push(pParam);
45.        }
46.
47.        return nParams.join("&");
48.    }
49.
50.    function saveResult(pMessage){
51.        var divStatus = document.getElementByld("divPostinfo");
52.        divStatus.innerHTML = "Server ได้รับ" +pMessage + "แล้ว";
53.    }
54.
55.    </script>
56. </head>
57. <body>
58. <form method= "post" action = "ResponsePost.php"onsubmit= "startRequest();return
    false">
59. <p><b>ตัวอย่างการส่ง Request แบบ POST </b></p>
60. <p>ป้อนข้อความที่ต้องการส่งไปยังฝั่ง Server:<input type = "text" name= "datapost" value=
    ""/> </b></p>
61. <p><input type = "submit" value = "ส่ง Request"/></p>
62. </form>
63. <div id= "divPostinfo"></div>
64. </body>
65. </html>
```

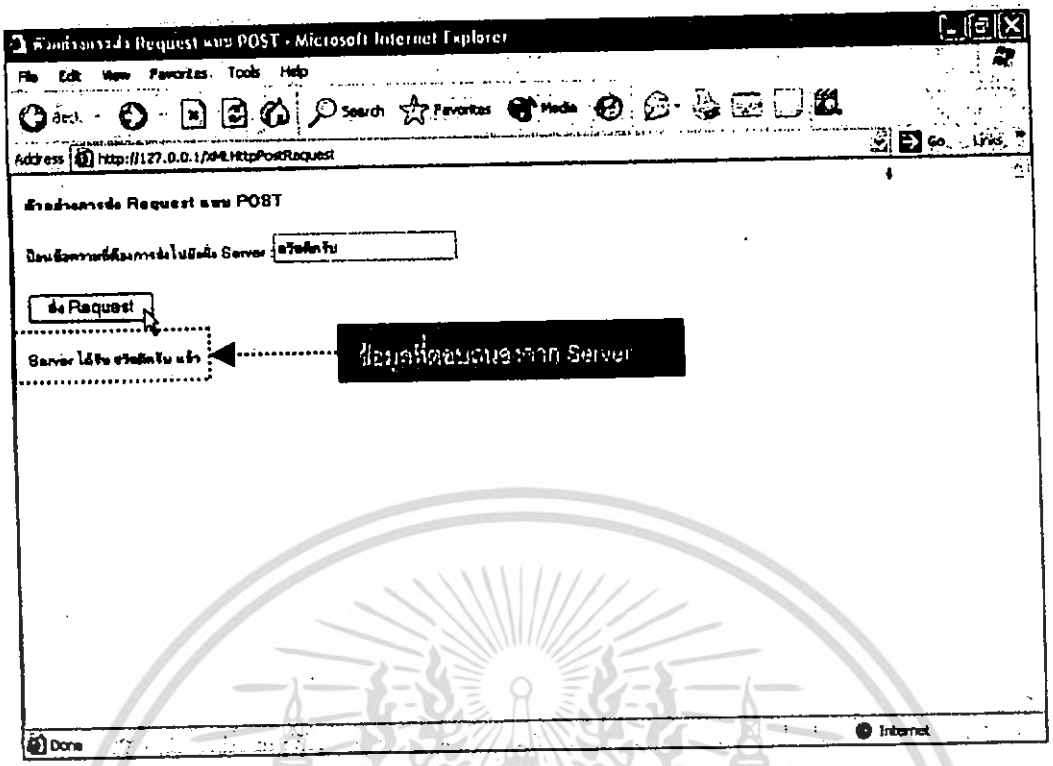
จากตัวอย่างที่ 2.7 ที่แสดงโค้ดของไฟล์ XMLHttpPostResponse.html การทำงานในโค้ดมี 3 ส่วนหลักๆที่สำคัญอย่างมากต่อการส่ง Request แบบ POST ได้แก่

บรรทัดที่ 58-62 เป็นกลุ่มโค้ดสำหรับสร้าง Form เพื่อรับข้อมูลจาก User

บรรทัดที่ 37-48 เป็นฟังก์ชันสำหรับนำข้อมูลจาก Form เข้ารหัสก่อนส่งไปยังฝั่ง Server

บรรทัดที่ 18-23 เป็นฟังก์ชันสำหรับส่ง Request ไปยังฝั่ง Server

โค้ดในการส่ง Request แบบ POST ที่ประกอบด้วยโค้ดทั้ง 3 ส่วนที่กล่าวมา การทำงานเริ่มจากเมื่อ User ป้อนข้อมูลผ่าน Form ที่อยู่บนเว็บเพจเสร็จแล้วคลิกปุ่ม "ส่ง Request" XMLHttpRequest จะเข้ารหัสข้อมูลและสร้าง Request แล้วส่ง Request ที่สร้างนั้นไปยังไฟล์ ResponsePost.php ที่ทำงานอยู่ฝั่ง Server จากนั้นฝั่ง Server จะส่ง Response กลับมายัง Web Browser ซึ่งมี XMLHttpRequest ทำหน้าที่รับ Response และส่งให้ JavaScript ทำหน้าที่ควบคุมการแสดงผลบนเว็บเพจต่อไป ดังแสดงให้เห็นในรูปที่ 2.25



รูปที่ 2.25 แสดงเว็บเพจที่ใช้โต้ตอบกับ User ของไฟล์ XMLHttpPostRequest.html

จากการใช้ XMLHttpRequest Objectดังตัวอย่างที่กล่าวมา จะเห็นได้ว่า XMLHttpRequest Object เป็นเทคโนโลยีที่เป็นตัวกลางที่สำคัญในการรับ-ส่ง Request ระหว่าง Web Browser กับ Server ซึ่งทำให้เว็บแอปพลิเคชั่นแบบ AJAX มีการจัดการกับการรับ-ส่ง Request แตกต่างจากเว็บแอปพลิเคชั่นแบบเดิม

2.3.4 Remote Scripting

Remote Scripting เป็นเทคโนโลยีที่ใช้โต้ตอบระหว่าง Web Browser กับ Server โดย Remote Scripting จะทำงานอยู่ที่ฝั่ง Server ประโยชน์ของ Remote Scripting คือ ช่วยลดการรีเฟรชหน้าจอ ซึ่งเหมือนกับการใช้ XMLHttpRequest Object ในเว็บแอปพลิเคชั่นแบบ AJAX สำหรับการนำ Remote Scripting ไปใช้งานสามารถแทรกรวมกับเทคโนโลยีอื่นๆได้ เช่น Flash Animation,Java Applet หรือ Active X เป็นต้น ในการเขียนโค้ดของ Remote Script สามารถใช้ IFRAME เพื่ออ้างถึงเอกสารอื่นๆได้ ดังจะแสดงให้เห็นในตัวอย่างต่อไปนี้

ตัวอย่างการใช้ Remote Script ที่แสดงต่อไปนี้เป็นการทำงานร่วมระหว่างภาษา JavaScript กับ IFRAME โดยเมื่อประมวลผลเสร็จเรียบร้อยแล้ว Server จะส่งผลลัพธ์ไปยัง Web Browser ดังตัวอย่างที่ 2.8

ตัวอย่างที่ 2.8 แสดงตัวอย่าง Remote Script กับ IFRAME ของไฟล์ Remote ScriptAndiframe.html

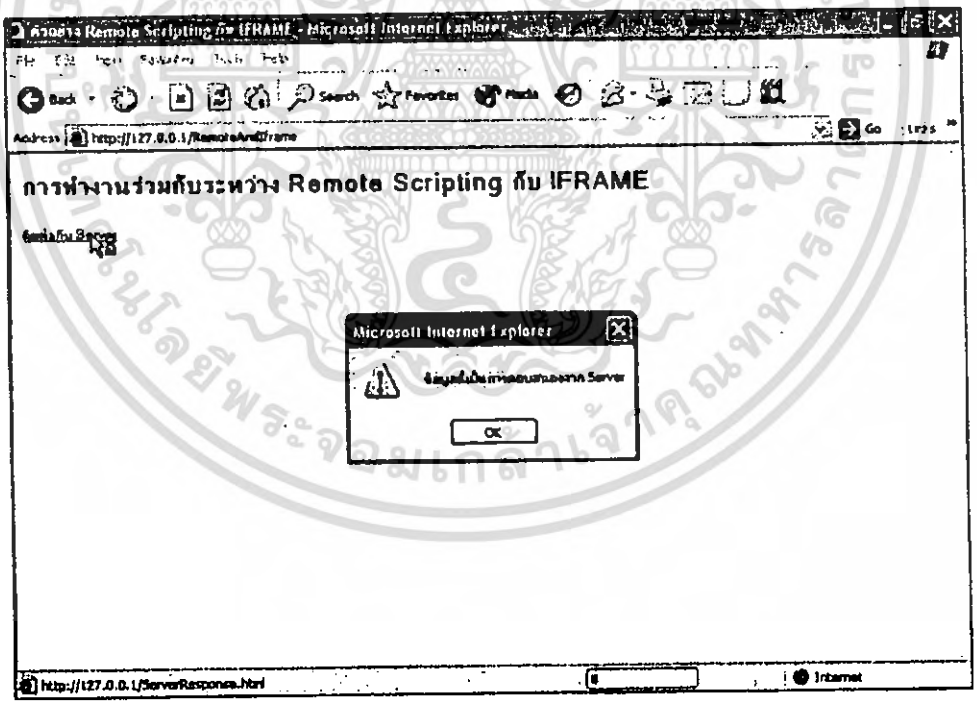
```
1. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2. <html>
3. <head>
4. <title>ตัวอย่าง Remote Scripting กับ IFRAME </title>
5. </head>
```

```
6.  <script type = "text/javascript">
7.
8.  function handleResponse(){
9.    alert('ข้อมูลนี้เป็นการตอบสนองจาก Server');
10. }
11. </script>
12. <body>
13. <h1>การทำงานร่วมกับระหว่าง Remote Scripting กับ IFRAME</h1>
14. <iframe id = "beforexhr" name = "beforexhr" style= "width:0px;height:0px;border:0px"
15. src= "blank.html"></iframe>
16. <a href= "server.html"target= "beforexhr">ติดต่อกับ Server</a>
17. </body>
18. </html>
```

ตัวอย่างที่ 2.9 แสดงโค้ดของไฟล์ ServerResponse.html

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2.  <html>
3.  <head>
4.  <title>ข้อมูลจากฝั่ง Server</title>
5.  </head>
6.  <script type = "text/javascript">
7.  window.parent.handleResponse();
8.  </script>
9.  <body>
10. </body>
11. </html>
```

จากตัวอย่างที่ 2.8 และ 2.9 เมื่อนำไฟล์ทั้งสองมาทดสอบการทำงานและแสดงผลด้วย Web Browser จะได้ผลลัพธ์การโต้ตอบกับ User ดังรูปที่ 2.26



รูปที่ 2.26 แสดงการโต้ตอบกับ User ของไฟล์ RemoteAndiframe.html

จากที่กล่าวมาถึงแม้ว่า Remote Scripting จะสามารถช่วยลดการรีเฟรชของหน้าจอได้ แต่การนำไปใช้งานจะแตกต่างจากการใช้ XMLHttpRequest Object ในเว็บแอปพลิเคชั่นแบบ AJAX โดย Remote Scripting จะถูกนำไปใช้งานที่ฝั่ง Server แต่ XMLHttpRequest Object จะถูกนำไปใช้งานที่ฝั่งของ Web Browser ซึ่งทำให้ทำงานได้อย่างรวดเร็วเนื่องจากการตอบสนองบางอย่างสามารถทำได้ทันทีโดยไม่ต้องติดต่อกับ Server ดังที่กล่าวไว้

2.3.5 ข้อดีและข้อเสียของ XMLHttpRequest Object

จากเนื้อหาที่กล่าวมาแล้ว ทำให้ทราบถึงการทำงาน XMLHttpRequest Object ในการสร้างและส่ง Request จาก Web Browser ไปยังฝั่ง Server ของเว็บแอปพลิเคชั่นแบบ AJAX ซึ่งสามารถสรุปข้อดีและข้อเสียของ XMLHttpRequest Object ได้ดังนี้

ข้อดี

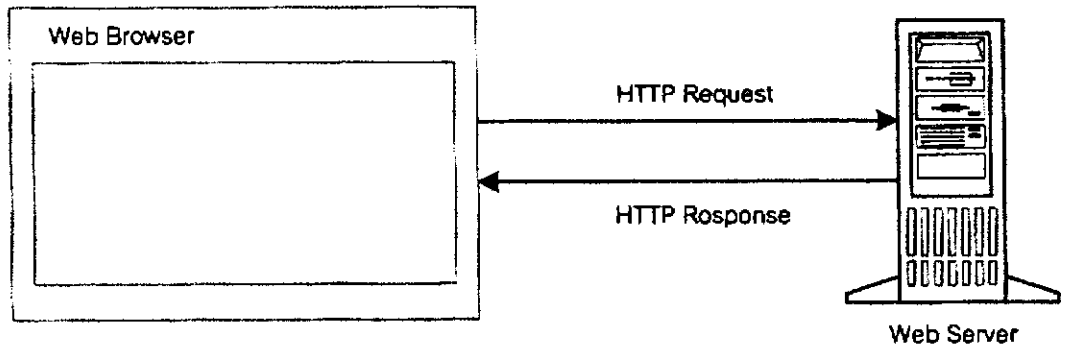
- โค้ดมีความเป็นระเบียบ แสดงขั้นตอนการทำงานอย่างชัดเจน ง่ายต่อการแก้ไข
- สามารถจัดการการทำงานที่ซับซ้อนได้ด้วยการเขียนโค้ดสั้นๆ
- มีรหัสแสดงสถานการณ์ทำงาน ทำให้ทราบสถานการณ์ทำงานในขั้นตอนต่างๆ

ข้อเสีย

สำหรับข้อเสียในการใช้ XMLHttpRequest Object เกิดขึ้นเมื่อ Internet Explorer มีการกำหนดระบบรักษาความปลอดภัยที่ไม่อนุญาตให้ ActiveX Object ทำงานได้ ส่งผลให้ XMLHttpRequest Object ไม่สามารถตรวจสอบว่า Internet Explorer นั้นใช้ ActiveX Object หรือไม่ และอาจมีปัญหาในการแสดงผลข้อมูลได้

2.3.6 การโอนถ่ายข้อมูลด้วย HTTP

Hypertext Transmission Protocol(HTTP) เป็นเทคโนโลยีที่นิยมใช้ในการนำมาโอนถ่ายข้อมูลผ่านอินเตอร์เน็ตในปัจจุบัน การทำงานของ HTTP มีสองส่วน ได้แก่ การโอนถ่าย Request และการโอนถ่าย Response การทำงานคร่าวๆเริ่มจาก User ร้องขอข้อมูลผ่าน Web Browser จากนั้น Web Browser จะสร้าง Request และส่งไปยัง Web Server โดยอาศัย HTTP และเมื่อ Web Server รับ Request แล้วจะประมวลผลและส่ง Response กลับมายัง Web Browser โดยอาศัย HTTP อีกครั้ง เมื่อ Web Browser รับ Response แล้วจึงนำข้อมูลมาจัดรูปแบบและแสดงข้อมูลบนเว็บเพจให้ User ได้เห็นต่อไป ดังรูป 2.27



รูปที่ 2.27 แสดงแบบจำลองการ ไอนถ่าย Request และ Response ของ HTTP

จากรูปที่ 2.27 แสดงการ ไอนถ่าย Request จากฝั่งWeb Browser ไปยังฝั่ง Web Server และไอนถ่าย Response จากฝั่ง Web Server กลับมายังฝั่ง Web Browser โดยอาศัย HTTP ไม่ว่าจะเป็นเว็บแอปพลิเคชั่นแบบเดิมหรือแบบ AJAX ต่างก็อาศัยการ ไอนถ่าย Request และ Response ด้วย HTTP เหมือนกัน แต่เมื่อพิจารณาด้านการงานแล้วเว็บแอปพลิเคชั่นแบบเดิมใช้ HTTP ไอนถ่ายข้อมูลบางช่วงเวลาเท่านั้น ขึ้นอยู่กับการร้องขอของ User ส่วนเว็บแอปพลิเคชั่นแบบ AJAX ใช้ HTTP ไอนถ่ายข้อมูลตลอดเวลา ดังจะเห็นได้จากการทำงานแบบ Asynchronous ของเว็บแอปพลิเคชั่นแบบ AJAX ตามที่กล่าวไว้โดยมีเทคโนโลยีที่เรียกว่า "XMLHttpRequest Object" เป็นตัวช่วยสร้างและส่ง Request ตามที่กล่าวมา

เทคนิคการจัดการกับ Response

จากที่กล่าวมาเกี่ยวกับเอกสารหรือข้อมูลที่ใช้ในการแลกเปลี่ยนกันระหว่างฝั่ง Web Browser กับฝั่ง Server ของเว็บแอปพลิเคชั่นแบบ AJAX นั้น อาจเป็นข้อความสั้นๆหรือเป็นเอกสาร XML ดังนั้น XMLHttpRequest Object จะมีการเรียกใช้ Properties ที่เกี่ยวข้องกับการจัดการกับ Response ที่มาจาก Server ทั้งในรูปแบบของข้อความและเอกสาร XML ดังนี้

- Properties ที่ชื่อ "responseText"ทำหน้าที่แสดงให้ทราบว่า Response ที่ส่งมาจาก Server เป็นข้อความ
- Properties ที่ชื่อ "responseXML"ทำหน้าที่แสดงให้ทราบว่า Response ที่ส่งมาจาก Server เป็นเอกสาร XML

Properties ข้างต้นทำหน้าที่แสดงสถานะของ Response ที่จะถูกส่งมาจากฝั่ง Server โดยจะใช้ JavaScript เขียนโค้ดเพื่อควบคุมการเรียกใช้งาน ส่งผลให้สามารถจัดการกับรูปแบบของ Response ทั้งที่เป็นข้อความและเอกสาร XML ได้อย่างเหมาะสม

การจัดการ Response ที่เป็น Simple Text

สำหรับ Response ที่ข้อความสั้นๆในที่นี้จะเรียกว่า "Simple Text" เช่น เมื่อ User ร้องขอข้อมูลแล้วได้รับข้อความสั้นๆแจ้งว่า "สำเร็จ" หรือ "ไม่สำเร็จ" เป็นต้น ซึ่งในที่นี้ได้แสดงตัวอย่างการเรียกใช้งานให้เห็นไปแล้วในตัวอย่างที่ 2.2 บรรทัดที่ 25

เนื่องจาก Response ที่เป็น Simple Text มีโครงสร้างที่ไม่แน่นอนและยากในการจัดการด้วย JavaScript คือ หากข้อความสั้นๆหลายข้อความ การใช้ JavaScript เขียนโค้ดเพื่อนำข้อความเหล่านั้นมาแสดงผลร่วมกันจะต้องใช้โค้ดจำนวนมากและมีการทำงานค่อนข้างซับซ้อน ส่งผลให้การสร้างเว็บเพจแบบ Dynamic เป็นไปได้ยาก ดังนั้นจึงได้มีการเรียกใช้ Properties ชื่อ "responseText" ร่วมกับ Properties ชื่อ "innerHTML" โดยขั้นแรกต้องจัดรูปแบบของ Simple Txet ให้มีความเป็นระเบียบแล้วเก็บไว้ที่ไฟล์ XML จากนั้นจึงสร้างไฟล์ HTML ที่เขียนโค้ดเพียงสั้นๆ เพื่อดึงข้อมูลที่จัดไว้นั้นมาแสดงบนเว็บเพจทำให้สามารถแสดง Response ที่ Simple Text ได้อย่างเป็นระเบียบ ดังตัวอย่างต่อไปนี้

ตัวอย่างการจัดการ Response ที่เป็น Simple Text

การใช้ Properties "responseText" ร่วมกับ "innerHTML" ที่จะแสดงให้เห็นต่อไปนี้เป็นตัวอย่างของเว็บแอปพลิเคชั่นแบบ AJAX ที่ใช้ XMLHttpRequest ในการส่ง Request แบบ GET โดยใช้ไฟล์ HTML สร้างเว็บเพจที่ใช้โต้ตอบกับ User และสร้างเอกสาร XML ที่มีข้อมูลเพื่อตอบสนอง User โดยเก็บเอกสาร XML ดังกล่าวไว้ที่ฝั่ง Server สำหรับการทำงานทั้งหมดมีขั้นตอนดังนี้

ขั้นตอนแรกต้องสร้างเอกสาร XML ที่ใช้ตอบสนอง User ในที่นี้ใช้ชื่อ "innerHtml.xml" ซึ่งมีการจัดรูปแบบข้อมูลไว้อย่างเป็นระเบียบ

ตัวอย่างที่ 2.10 แสดงโค้ดไฟล์ InnerHtml.xml

```
1.  <?xml version= "1.0" encoding= "tis-620"?>
2.  <table border= "1">
3.        <tbody>
4.              <tr>
5.                    <th>Customer Name</th>
6.                    <th>Location</th>
7.                    <th>E-mail</th>
8.              </tr>
9.              <tr>
10.                  <td>Saksit  maharat</td>
11.                  <td>Ladkrabang</td>
12.                  <td>Mahasaksit@hotmail.com</td>
13.             </tr>
14.             <tr>
15.                   <td>Aungkana wongwa</td>
16.                   <td>Bangna</td>
17.                   <td>A1234@hotmail.com</td>
18.             </tr>
19.       </tbody>
20. </table>
```

เมื่อสร้างเอกสาร XML และเก็บไว้ที่ฝั่ง Server เรียบร้อยแล้ว ขั้นตอนต่อไปต้องสร้างไฟล์ HTML เพื่อใช้โต้ตอบกับ User ในที่นี้ชื่อ "SimpleInnerHtml.html"

ตัวอย่างที่ 2.11 แสดงโค้ดการใช้ responseText และ innerHTML ของไฟล์ SimpleInnerHtml.html

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
2.  http://www.w3.org/TR/xhtml1-strict.dtd>
3.  <html xmlns= "http://www.w3.org/1999/xhtml">
4.  <head>
5.  <title>ตัวอย่างการเรียกใช้ responseText กับ innerHTML</title>
6.   <script type = "text/javascript">
7.  var xmlHttp;
8.  function createXMLHttpRequest(){
9.      if (window.ActiveXObject){
10.        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
11. }
12.     else if (window.XMLHttpRequest){
13.        xmlHttp = new XMLHttpRequest();
14.     }
15. }
16. function startRequest(){
17.      createXMLHttpRequest();
18.      xmlHttp.onreadystatechange = handleStateChange;
19.      xmlHttp.open("GET", "InnerHTML.xml",true);
20.      xmlHttp.send(null);
21. }
22. function handleStateChange(){
23.      if(xmlHttp.readyState == 4){
24.        if(xmlHttp.status == 200){
25.          document.getElementById("results").innerHTML = xmlHttp.responseText;
26.      }
27.    }
28. }
29. </script>
30. </head>
31. <body>
32.      <form action= "#">
33.        <input type= "button" value= "คลิกเพื่อแสดงข้อมูล "onclick= "startRequest();"/>
34. </from>
35. <div id= "results"></div>
36. </body>
37. </html>
```

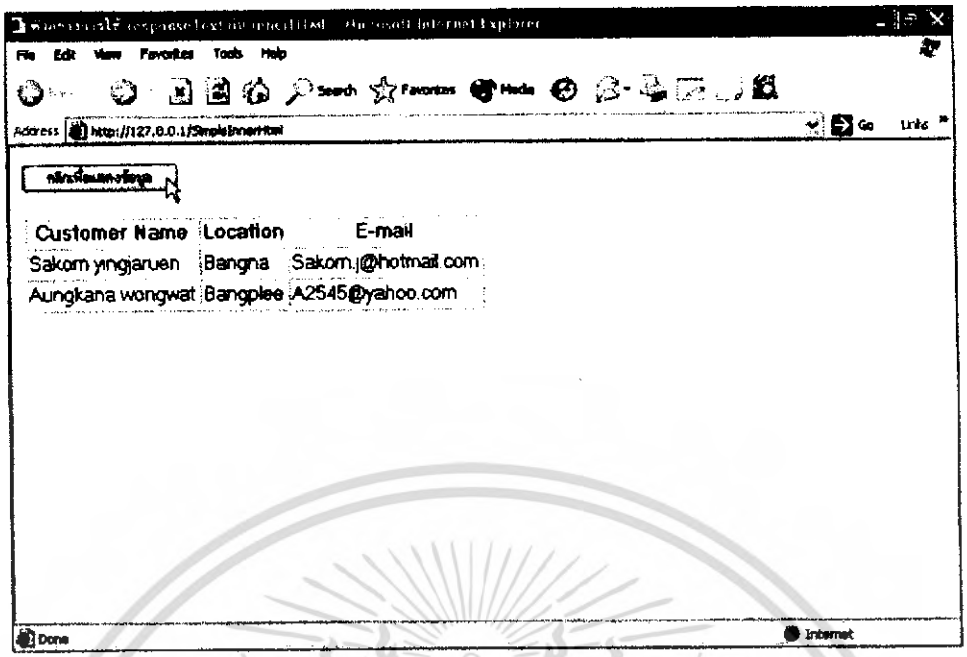
จากตัวอย่างที่ 2.11 แสดงโค้ดของไฟล์ SimpleInnerHtml.html โดยมีโค้ดในการเรียกใช้ responseText และ innerHTML ดังคำสั่งในบรรทัดที่ 25 ดังนี้

```
Document.getElementById("results").innerHTML = xmlHttp.responseText;
```

จากคำสั่งดังกล่าวจะทำให้ทราบสถานะของ Response ที่เป็น Simple Text ซึ่งอยู่ในไฟล์ InnerHTML.xml โดยสามารถร้องขอแบบ GET ดังคำสั่งในบรรทัดที่ 19 ดังนี้

```
xmlHttp.open("GET", "InnerHTML.xml", true);
```

เมื่อสร้างไฟล์ SimpleInnerHtml.html และ InnerHTML.xml เสร็จแล้วสามารถทดสอบการใช้งานด้วย Web Browser ได้หลายรุ่นดังที่กล่าวมาแล้วว่ามีรุ่นใดบ้างที่สนับสนุนการใช้งาน AJAX ซึ่งในที่นี้ใช้ Internet Explorer เวอร์ชัน 6.0 สามารถแสดงผลได้ดังรูปที่ 2.27



รูปที่ 2.28 แสดงการโต้ตอบกับ User ไฟล์ SimpleInnerHtml.html

จากรูปที่ 2.28 แสดงการโต้ตอบกับ User ของไฟล์ SimpleInnerHttp.html โดยมีปุ่มที่ให้ User ใช้คลิกเพื่อร้องขอข้อมูล คือ ปุ่ม "คลิกเพื่อแสดงข้อมูล" เมื่อ User คลิกเมาส์ที่ปุ่มดังกล่าว ไฟล์ SimpleInnerHtml.html จะเรียกใช้ Properties ชื่อ "responseText" และ "innerHTML" โดยโค้ดของ JavaScript ที่เขียนเพียงสั้นๆเท่านั้น ดังโค้ดในตัวอย่างที่ 3.1 บรรทัดที่ 25 โค้ดดังกล่าวจะดึงข้อมูลในไฟล์ InnerHTML.xml(อยู่ที่ฝั่ง Server) มาแสดงผลบนเว็บเพจซึ่งถูกจัดรูปแบบข้อมูลไว้เรียบร้อยแล้ว ทำให้ผลลัพธ์ที่แสดงให้ User เห็นนั้นมีความเป็นระเบียบเรียบร้อย จากการทำงานที่กล่าวมาถึงแม้การจัดการกับ Response ที่เป็น Simple Text จะใช้โค้ดเพียงสั้นๆก็ตาม แต่สิ่งสำคัญคือรูปแบบของข้อมูลที่อยู่ในไฟล์ XML ที่จะนำมาแสดงผลนั้นต้องมีการจัดการที่เป็นระเบียบไม่เช่นนั้นอาจส่งผลต่อการแสดงผลข้อมูลได้

การจัดการ Response ที่เป็นเอกสาร XML

นอกจาก Response ที่ส่งมาจากฝั่ง Server จะมีรูปแบบเป็น Simple Text ตามที่กล่าวในหัวข้อที่ผ่านมาแล้ว บางครั้ง Response อาจอยู่ในรูปแบบของเอกสาร XML ซึ่ง XMLHttpRequest Object ต้องเรียก Properties ชื่อ "responseXML" ที่ทำหน้าที่แสดงสถานะของ Response ที่เป็นเอกสาร XML มาใช้งานด้วย และนอกจาก ResponseXML แล้วการจัดการ Response ที่เป็นเอกสาร XML ต้องอาศัย Methods และ Properties ที่เป็นส่วนประกอบของเทคโนโลยีที่เรียกว่า "DOM"(มีหน้าที่จัดการโครงสร้างและรูปแบบของเอกสารต่างๆตามที่กล่าวมาแล้ว)โดย Methods และ Properties ที่เรียกมาใช้แสดงไว้ในตารางที่ 2.3 และ 2.4

ตารางที่ 2.3 แสดง Methods ที่เป็นส่วนประกอบของ DOM สำหรับจัดการเอกสาร XML

ชื่อ Methods	คำอธิบาย
getElementByID(id)(document)	รับ Element จาก Document เป็น ID
getElementByTagName(name)	รับ Element เป็น name
hasChildNodes()	ใช้กำหนดหรือคืนค่า Boolean ถ้าโหนดนั้นมี Child node จะคืนค่า True ถ้าไม่มีจะคืนค่า False
getAttribute(name)	รับ Attribute เป็น name

ตารางที่ 2.4 แสดง Properties ที่เป็นส่วนประกอบของ DOM สำหรับจัดการเอกสาร XML

ชื่อ Properties	คำอธิบาย
childNodes	ใช้อ่านอย่างเดียวโดยที่เป็นที่เก็บรวบรวมรายการของ Element ที่มี Children หลายตัว
firstChild	ใช้คืนค่าโหนดที่เป็น Child ตำแหน่งแรกออกมา ถ้าไม่มีก็จะคืนค่า Null มาแทน
lastChid	ใช้คืนค่าโหนดที่เป็น Child ตำแหน่งสุดท้ายออกมา ถ้าไม่มีก็จะคืนค่า Null มาแทน
nextSibling	ใช้ตั้งค่าหรือใช้คืนค่าโหนดที่เป็น Child ตำแหน่งต่อไปที่อยู่ใน Parent เดียวกัน
nodeValue	จะใช้กำหนดหรือคืนค่าของโหนด โดยขึ้นอยู่กับชนิดของข้อมูลด้วย
parentNode	ใช้คืนค่าที่เป็น Parent ของทุกโหนด ยกเว้น Document,DocumentFragment และ Attribute ที่ไม่มีโหนด Parent
previousSibiling	ใช้คืนค่าโหนด Child ที่ตำแหน่งก่อนหน้าโหนดปัจจุบันใน Parent เดียวกัน

จากตารางที่ 2.3 และ 2.4 แสดงให้เห็น Methods และ Properties ที่เป็นส่วนประกอบของ DOM ที่ใช้จัดการเอกสาร XML สำหรับการเรียกใช้ Methods และ Properties ตามที่กล่าวมานั้นจะต้องเขียนโค้ดด้วย JavaScript เพื่อกำหนดขั้นตอนการทำงานต่างๆดังตัวอย่างต่อไปนี้

ตัวอย่างการจัดการ Response ที่เป็นเอกสาร XML

ตัวอย่างที่จะกล่าวดังต่อไปนี้เป็นการจัดการ Response ที่เป็นเอกสาร XML โดยใช้ JavaScript เขียนโค้ดควบคุมการรับ Response ซึ่งขั้นตอนแรกต้องสร้างเอกสาร XML เก็บไว้ที่ Server ในที่นี้ใช้ชื่อ "ResponseXML.xml"

ตัวอย่างที่ 2.12 แสดงโค้ดของไฟล์ ResponseXML.xml

```
1. <?xml version= "1.0"encoding= "tis-620"?>
2. <provinces>
3.     <north>
4.         <province>เชียงใหม่</province>
```

```
5.          <province>เชียงราย</province>
6.          <province>แม่ฮ่องสอน</province>
7.       </north>
8.       <south>
9.          <province>ภูเก็ต</province>
10.         <province>สงขลา</province>
11.         <province>กระบี่</province>
12.      </south>
13.      <east>
14.         <province>ระยอง</province>
15.         <province>ชลบุรี</province>
16.         <province>ตราด</province>
17.      </east>
18.      <west>
19.         <province>กาญจนบุรี</province>
20.         <province>ตาก</province>
21.         <province>เพชรบุรี</province>
22.      </west>
23. </provinces>
```

เมื่อสร้างเอกสาร XML เก็บไว้ที่ฝั่ง Server เรียบร้อยแล้ว ขั้นตอนที่ 2 เป็นการสร้างไฟล์ HTML เพื่อใช้เป็นเว็บเพจที่โต้ตอบกับ User ในที่นี้ใช้ไฟล์ HTML ชื่อ "SimpleParseXML.html"

ตัวอย่างที่ 2.13 แสดงโค้ดของไฟล์ SimpleParseXML.html

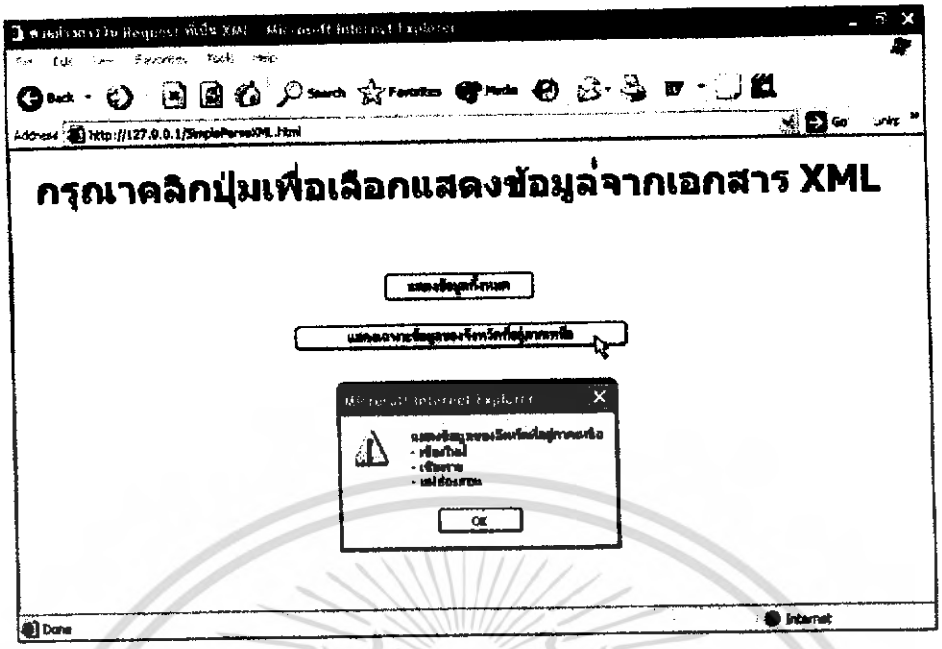
```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
2.  http://www.w3.org/TR/xhtml1-strict.dtd>
3.  <html xmlns= "http://www.w3.org/1999/xhtml">
4.  <head>
5.  <title>ตัวอย่างการรับ Request ที่เป็น XML </title>
6.  <script type = "text/javascript">
7.  var xmlHttp;
8.  ver requestType = "";
9.
10. function createXMLHttpRequest(){
11.     if (window.ActiveXObject){
12.       xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
13. }
14.     else if (window.XMLHttpRequest){
15.       xmlHttp = new XMLHttpRequest();
16.     }
17. }
18.
19. function startRequest(requestedList){
20.     requestType=requesttedList;
21.     createXMLHttpRequest();
22.     xmlHttp.onreadystatechange = handleProvincesChange;
23.     xmlHttp.open("GET", "ResponseXML.xml",true);
24.     xmlHttp.send(null);
25. }
26. function handleProvincesChange(){
27.     if(xmlHttp.readyState == 4){
28.       if(xmlHttp.status == 200){
29.         if(requestType == "north"){
30.           listNorthProvinces();
31.         }
32.         else if(requestType == "all"){
33.           listAllProvinces();
34.     }
35.   }
```

```
36. }
37. }
38.
39. function listNorthProvinces(){
40.         var xmlDoc = xmlHttp.responseXML;
41.         var northNode = xmlDoc.getElementsByTagName("north")[0];
42.
43.         var out = "Northern Province";
44.         var northProvinces = northNode.getElementByTagName("state");
45.
46.         outputList("แสดงข้อมูลของจังหวัดที่อยู่ภาคเหนือ",northProvinces);
47. }
48.
49. function listAllProvinces(){
50.         var xmlDoc = xmlHttp.responseXML;
51.         var  allProvinces = xmlDoc.getElementsByTagName("Province");
52.
53.         outputList("แสดงข้อมูลทั้งหมด", allProvinces);
54. }
55. function outputList(title,provinces){
56.          var out = title;
57.          var currentProvinces = null;
58.          for(var I =0;I < province.length;i++){
59.                currentProvinces = provinces[i];
60.                out = out + "\n-" +currentProvinces.childNodes[0].nodeValue;
61.           }
62.
63.          alert(out);
64. }
65. </script>
66. </head>
67.
68. <body>
69.      <center>
70. <h1>กรุณาคลิกปุ่มเพื่อเลือกแสดงข้อมูลจากเอกสาร XML </h1>
71.      <br/>
72. <br/>
73.      <form action = "#">
74.           <input type = "button" value= "แสดงข้อมูลทั้งหมด" onclick= "startRequest('all');"/>
75.          <br/><br/>
76.          <input type = "button" value= "แสดงข้อมูลเฉพาะจังหวัดที่อยู่ภาคเหนือ" onclick=
     "startRequest('north');"/>
77.          </form>
78. </center>
79. </body>
80. </html>
```

จากตัวอย่างที่ 2.13 แสดงโค้ดของไฟล์ SimpleParseXML.html ที่ใช้จัดการกับ Response ที่เป็นเอกสาร XML ที่ชื่อ "ResponseXML.xml" ที่อยู่ฝั่ง Server โดยเรียกใช้ Properties ชื่อ "responseXML" ดังแสดงในบรรทัดที่ 40 และ 50 ดังนี้

```
var xmlDoc = xmlHttp.responseXML;
```

เมื่อสร้างไฟล์ SimpleParseXML.html ที่ใช้สำหรับโต้ตอบกับ User เสร็จแล้ว สามารถเรียกใช้ Web Browser แสดงผลลัพธ์ที่เกิดจากโค้ดดังกล่าวได้ ในที่นี้ใช้ Internet Explorer ดังรูปที่ 2.28



รูปที่ 2.29 แสดงเว็บเพจของไฟล์ SimpleParseXML.html

จากรูปที่ 2.29 แสดงเว็บเพจที่ใช้สำหรับโต้ตอบกับ User โดยมีปุ่ม 2 ปุ่ม สำหรับปุ่มแรกชื่อ "แสดงข้อมูลทั้งหมด" ใช้ร้องขอข้อมูลจากเอกสาร XML ชื่อ ResponseXML.xml" มาแสดงทั้งหมด และอีกปุ่มชื่อ "แสดงเฉพาะข้อมูลของจังหวัดที่อยู่ภาคเหนือ" เพื่อใช้แสดงข้อมูลเพียวบางส่วนเท่านั้น เมื่อ User คลิกปุ่มใดปุ่มหนึ่งบนหน้าจอจะปรากฏกล่องข้อความแสดงข้อมูลตามที่ User ต้องการ

เทคนิคการส่ง Request

การส่ง Request จากฝั่ง Web Browser ไปยังฝั่ง Server นั้น เป็นการส่งค่าพารามิเตอร์บางอย่างไปยังฝั่ง Server จากนั้น Server จะนำค่าพารามิเตอร์ไปประมวลผลและส่ง Response กลับมายังฝั่ง Web Browser เช่น การป้อนข้อมูลบนเว็บเพจ โดย User ป้อนข้อมูลที่เป็นรหัสลูกค้า แล้วมีชื่อลูกค้าปรากฏขึ้นมาบนเว็บเพจลักษณะดังกล่าวจะถือว่า รหัสลูกค้า คือค่าพารามิเตอร์ เป็นต้น สำหรับการส่ง Request ดังกล่าวเป็นหน้าที่ของ XMLHttpRequest Object ตามที่กล่าวมาแล้วโดนสามารถส่ง Request ได้ทั้ง 2 แบบ คือ ทั้งแบบ GET และ POST แสดงให้เห็นการทำงานดังตัวอย่างต่อไปนี้

ส่ง Request ที่มีค่า Parameters โดยใช้การส่งแบบ GET

ตัวอย่างการส่ง Request ต่อไปนี้เป็นการใช้งาน AJAX ร่วมกับภาษา PHP(อาจใช้ภาษาอื่นก็ได้) ซึ่ง AJAX ใช้ XMLHttpRequest Object ช่วยส่ง Request จาก Web Browser ไปยังฝั่ง Server เพื่อร้องขอข้อมูลที่เก็บอยู่ในฐานข้อมูลในที่นี้ใช้ฐานข้อมูลของ MySQL (เมื่อใช้งานสามารถใช้ฐานข้อมูลอื่นได้ และโค้ดที่ใช้จะแตกต่างกันด้วย) โดยใน Request นั้นจะส่ง Parameter ไปด้วย

ขั้นตอนแรกต้องสร้างฐานข้อมูลไว้ทางฝั่งของ Server ก่อน ในตัวอย่างนี้จะใช้ฐานข้อมูลชื่อ "My_Database" และมีตารางข้อมูลชื่อ "Customers" ซึ่งมีโครงสร้างดังตัวอย่างที่ 3.5

ตัวอย่างที่ 2.14 แสดงโครงสร้างและข้อมูลของตาราง Customers

```
#โครงสร้างข้อมูลสำหรับตาราง 'Customers'

CREATE TABLE 'Customers '(
'CustomerId' int(11) NOT NULL auto_increment,
'Name' varchar(255) NOT NULL default'',
'Address' varchar(255) NOT NULL default'',
PRIMARY KEY ('CustomerId')
)TYPE=MyISAM COMMENT= 'ตัวอย่างข้อมูลลูกค้า';

INSERT INTO 'Customers' VALUES(1, 'Somsrak Hanprasert, '321 Prachautid Street');
INSERT INTO 'Customers' VALUES(2, 'Somjai  Kuertdee, '652 Taksin Road');
INSERT INTO 'Customers' VALUES(3, 'Doungnert Wongcharuan, '20 Klonggoa');
INSERT INTO 'Customers' VALUES(4, 'Aphichat Panphom, '125 Sukumwit Road');
```

จากตารางข้อมูลตัวอย่างที่ 2.14 แสดงให้เห็นโครงสร้างและข้อมูลของตาราง Customers ซึ่งเป็นเพียงตัวอย่างง่ายๆเท่านั้น เมื่อนำไปใช้จริงต่อไป ผู้ใช้สามารถปรับปรุงหรือเพิ่มข้อมูลให้เหมาะกับการใช้งานได้ และเมื่อมีฐานข้อมูลตามตัวอย่างข้างต้นแล้ว ขั้นตอนต่อไปต้องสร้างไฟล์ด้วยภาษา PHP ในที่นี้ใช้ชื่อ "GetCustomerData.php"

ตัวอย่างที่ 2.15 แสดงโค้ดของไฟล์ GetCustomerData.php

```
1.  <?php
2.  header("Content-Type:text/plain; charset=TIS-620");
3.
4.  //รับข้อมูลจาก id เก็บไว้ในตัวแปร $clD
5.  $clD = $_GET["id"];
6.
7.  //กำหนดตัวแปรที่เก็บผลลัพธ์
8.  $info = "";
9.
10. //ข้อมูลเกี่ยวกับฐานข้อมูลที่เรียกใช้
11. $DBServer = "localhost";
12. $DBName = "My_Database";
13. $DBUsername = "";
14. $DBPassword = "";
15.
16. //คำสั่งดึงข้อมูล
17. $Query = "Select * from Customers where CustomerId =".$clD;
18.
19. //เชื่อมต่อฐานข้อมูล
20. $Link = mysql_connect($DBServer,$DBusername,$DBPassword);
21. @mysql_select_db($DBName) or $sInfo = "ไม่สามารถเข้าถึงฐานข้อมูลได้";
22.
23. if($Result = mysql_query($Query) and mysql_num_rows($Result) > 0){
24.     $Values = mysql_fetch_array($Result,MYSQL_ASSOC);
25.     $Info = $Values['Name']. "<br/>".$Values['Address']. "<br/>";
26. }else{
27.     $Info = "ไม่มีข้อมูลลูกค้าที่มีรหัสเป็น $clD";
28. }
29.
30. mysql_close($Link);
31. echo $Info;
32. ?>
```

จากตัวอย่างที่ 2.15 แสดงโค้ดของไฟล์ GetCustomerData.php ซึ่งเขียนด้วยภาษา PHP (ทำงานที่ฝั่ง Server) มีหน้าที่รับ Request ที่ส่งมาจาก XMLHttpRequest Object นอกจากนี้ไฟล์


```
3.  <head>
```

GetCustomerData.php ยังทำหน้าที่เชื่อมต่อกับฐานข้อมูลชื่อ "My_Database" ที่สร้างไว้แล้วตามตัวอย่างที่ 2.14 จากโค้ดดังกล่าวสังเกตว่าไม่มีโค้ดของ HTML และ JavaScript รวมอยู่เลยเนื่องจากโค้ดของ HTML และ JavaScript มีการนำไปใช้ในไฟล์ HTML ซึ่งจะแสดงให้เห็นในขั้นตอนต่อไป จึงทำให้สามารถจัดการกับโค้ดของไฟล์ GetCustomerData.php ได้ง่ายและแสดงขั้นตอนการทำงานของโค้ดได้อย่างชัดเจน

ขั้นตอนต่อไปนี้ คือ การสร้างเว็บเพจเพื่อให้ User ใช้ป้อนข้อมูล ในที่นี้เป็นไฟล์ HTML ชื่อ "GetRequestPar.html" ที่ใช้ Javascript ควบคุมการทำงาน

ตัวอย่างที่ 2.16 แสดงโค้ดของไฟล์ GetRequestPar.html

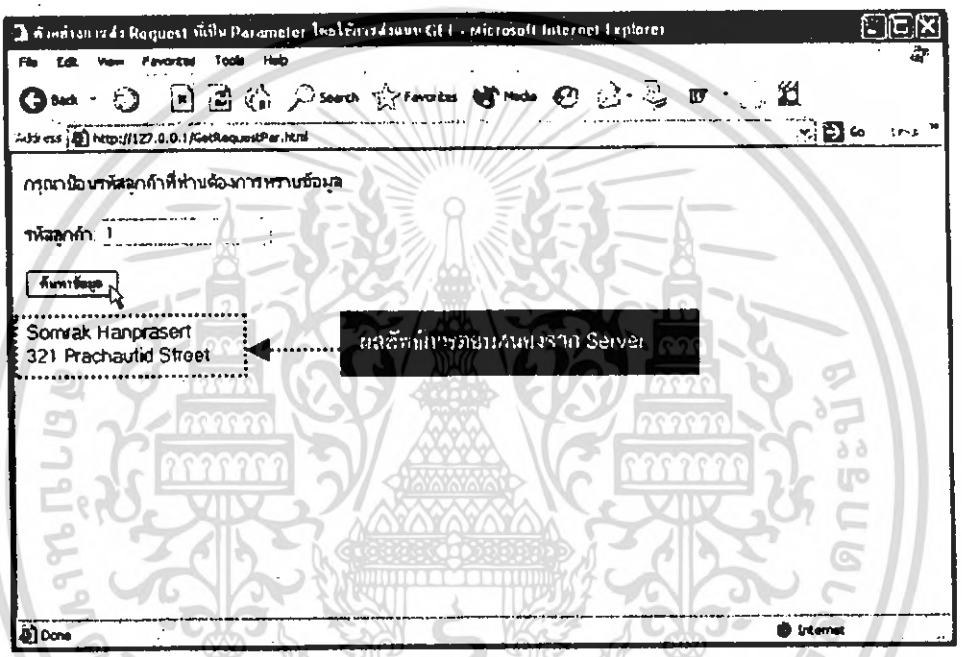
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<meta http-equiv= 'content-type' content= 'text/html' charset= 'windows-874'>
<title>ตัวอย่างการส่ง Request แบบ GET </title>

<script type = "text/javascript">
    var xmlHttp;

function createXMLHttpRequest(){
    if(window.ActiveXObject){
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    else if (window.XMLHttpRequest){
        xmlHttp = new XMLHttpRequest();
    }
}

function startRequest(){
    createXMLHttpRequest();
    var cId = document.getElementById("txtCustomerId").value;
    xmlHttp.open("get", "GetCustomerData.php?id = "+ cId,true);
    xmlHttp.onreadystatechange = function(){
      if(xmlHttp.readyState == 4){
        if(xmlHttp.status == 200){
            displayinfo(xmlHttp.responseText);
        }else{
            displayinfo("พบข้อผิดพลาด:"+xmlHttp.statusText);
        }
    }
};
    xmlHttp.send(null);
}

    function displayinfo(){
        document.getElementById( "divCustomerInfo").innerHTML =
    xmlHttp.responseText;

    }
</script>
</head>
<body>
<p>กรุณาป้อนรหัสลูกค้าที่ท่านต้องการทราบข้อมูลช</p>
<p>รหัสลูกค้า:<input type = "text" id= "txtCustomerId" value= ""/> </p>
<p><input type = "button" value = "ค้นหาข้อมูล" onClick= "startRequest()"/></p>
```

```
45. <div id= "divCustomerinfo"></div>
46. </body>
47. </html>
```

จากตัวอย่างที่ 2.16 ที่แสดงโค้ดของไฟล์ GetRequestPar.html โดยแสดงให้เห็นถึงรูปแบบการส่ง Request แบบ GET ซึ่งระบุค่าข้อมูลที่ User ต้องการผ่านทาง URL ซึ่งก็คือ การส่ง Parameters ไปด้วยนั่นเอง ตามคำสั่งในบรรทัดที่ 22 ดังนี้

```
oXmlHttp.open("get", "GetCustomerData.php?id= "+cId,true);
```

จากคำสั่งที่ส่ง Request แบบ GET ดังกล่าว ค่า Parameters ก็คือค่าที่อยู่ในตัวแปรชื่อ "cId" นั่นเอง และเมื่อ User ป้อนข้อมูลที่ต้องการผ่าน Web Browser แล้วจะได้รับการตอบสนองจาก Server ดังรูปที่ 2.30



รูปที่ 2.30 แสดงหน้าจอที่ใช้ในการโต้ตอบกับ User ของไฟล์ GetRequestPar.html

จากรูปที่ 2.30 แสดงการโต้ตอบกับ User ของไฟล์ GetRequestPar.html โดยมีกล่องข้อความสำหรับให้ User ป้อนรหัสของลูกค้าที่ User ต้องการทราบข้อมูล เมื่อป้อนข้อมูลและคลิกปุ่ม "ค้นหาข้อมูล" บนหน้าจอจะปรากฏข้อความที่เกิดจากการตอบสนองจากฝั่ง Server โดยการทำงานดังกล่าวเกิดจากการส่ง Request ที่มีค่าพารามิเตอร์และใช้การส่งแบบ GET นั่นเอง

การส่ง Request ที่มีค่า Parameter โดยใช้การส่งแบบ POST

จากตัวอย่างการใช้ XMLHttpRequest Object ส่งค่า Request แบบ GET ที่กล่าวมาเป็นการส่ง Request ผ่าน URL แต่การใช้ XMLHttpRequest Object ส่ง Request แบบ POST ที่จะกล่าวดังต่อไปนี้มีการใช้งานแตกต่างกัน คือ การส่ง Request แบบ POST จะใช้ Form เพื่อให้ User ป้อนข้อมูล จากนั้น XMLHttpRequest Object จึงนำข้อมูลดังกล่าวส่งไปยังฝั่ง Server ตามขั้นตอนต่อไปนี้

ขั้นแรกนั้นต้องมีฐานข้อมูลไว้ทางฝั่ง Server ในที่นี้ใช้ข้อมูลที่สร้างไว้แล้วตามตัวอย่างที่ 2.14 คือ "My_Database" และมีตารางชื่อ "Customers" จากนั้น สร้างไฟล์ที่ทำหน้าที่รับข้อมูลจาก Form ในที่นี้ใช้ภาษา PHP สร้างไฟล์ชื่อ "PostCustomerData.php" แสดงให้เห็นโค้ดได้ดังนี้

ตัวอย่างที่ 2.17 แสดงโค้ดของไฟล์ PostCustomerData.php

```
1. <?php
2.   header("Content-Type:text/plain; charset=TIS-620");
3.
4.   //รับข้อมูล
5.   $Name =$_POST["txtName"];
6.   $Address =$_POST["txtAddress"];
7.
8.   //กำหนดตัวแปรที่เก็บข้อมูลผลลัพธ์
9.   $Status = "";
10.
11. //ข้อมูลเกี่ยวกับฐานข้อมูลที่เรียกใช้
12. $DBServer = "localhost";
13. $DBName = "My_Database";
14. $DBUsername = "";
15. $DBPassword = "";
16.
17. //คำสั่งดึงข้อมูล
18. $SQL = "Insert into Customers(Name,Address) values ('$Name', '$Address')";
19.
20. //เชื่อมต่อฐานข้อมูล
21. $Link = mysql_connect($DBServer,$DBUsername,$DBPassword);
22. @mysql_select_db($DBName) or $Status = "ไม่สามารถเข้าถึงฐานข้อมูลได้";
23.
24. if($Result = mysql_query($SQL)){
25.     $Status = "รหัสลูกค้าที่ได้รับคือ ".mysql_insert_id();
26. }else{
27.     $Status = "มีข้อผิดพลาดเกิดขึ้น ไม่สามารถบันทึกข้อมูลได้";
28. }
29.
30. mysql_close($Link);
31. echo $Status;
32. ?>
```

จากตัวอย่างที่ 2.17 เป็นโค้ดที่เขียนด้วยภาษา PHP สำหรับรับค่าข้อมูลจาก Form ซึ่งค่าที่มาจาก Form จะถูกส่งไปจัดเก็บยังฐานข้อมูล โดยค่าที่ถูกส่งมาจาก Form นั้นมีการจัดการด้วยโค้ดของ JavaScript ที่อยู่ในไฟล์ HTML ชื่อ "PostRequestPar.html"

ตัวอย่างที่ 2.18 แสดงโค้ดของไฟล์ PostRequestPar.html

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"><html>
2.  <head>
3.  <meta http-equiv= 'content-type' content= 'text/html' charset= 'windows-874'>
4.  <title>ตัวอย่างการส่ง Request ที่เป็น Parameter โดยใช้การส่งแบบ POST</title>
5.   <script type = "text/javascript">
6.      var xmlHttp;
7.
8.  function createXMLHttpRequest(){
9.      if(window.ActiveXObject){
10.         xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
11.     }
12.     else if (window.XMLHttpRequest){
```

```
13.          xmlHttp = new XMLHttpRequest();
14.     }
15. }
16.
17. function startRequest(){
18.     createXMLHttpRequest();
19.     var pForm = document.forms[0];
20.     var pBody = getRequestBody(pForm);
21.     xmlHttp.open("post", pForm.action,true);
22.     xmlHttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
23.
24.     xmlHttp.onreadystatechange = function(){
25.       if(xmlHttp.readyState == 4){
26.          if(xmlHttp.status == 200){
27.              saveResult(xmlHttp.responseText);
28.       }else{
29.            saveResult("พบข้อผิดพลาด:"+xmlHttp.statusText);
30.       }
31.     }
32. );
33.     xmlHttp.send(pBody);
34. }
35.
36.    function getRequestBody(pForm){
37.         var nParams = new Array();
38.
39.         for(var i=0;i<pForm.elements.length;i++){
40.             var pParam = encodeURLComponent(pForm.elements[i].name);
41.             pParam += "=";
42.             pParam += encodeURLComponent(pForm.elements[i].value);
43.             nParams.push(pParam);
44.         }
45.
46.         return nParams.join("&");
47.     }
48.
49.     function saveResult(pMessage){
50.         var divStatus = document.getElementById("divPostinfo");
51.         divStatus.innerHTML = "บันทึกข้อมูลเสร็จแล้ว:" +pMessage;
52.     }
53.
54.    </script>
55. </head>
56. <body>
57. <form method= "post" action = "PostCustomerData.php"onsubmit=
    "startRequest();return false">
58. <p>ป้อนข้อมูลลูกค้าที่ต้องการบันทึก</p>
59. <p>ชื่อลูกค้า<input type = "text" name= "txtName" value= ""/> <br/>
60. <p>ที่อยู่ลูกค้า<input type = "text" name= "txtAddress" value= ""/> <br/>
61. <p><input type = "submit" value = "บันทึกข้อมูล"/></p>
62. </form>
63. <div id= "divPostinfo"></div>
64. </body>
65. </html>
```

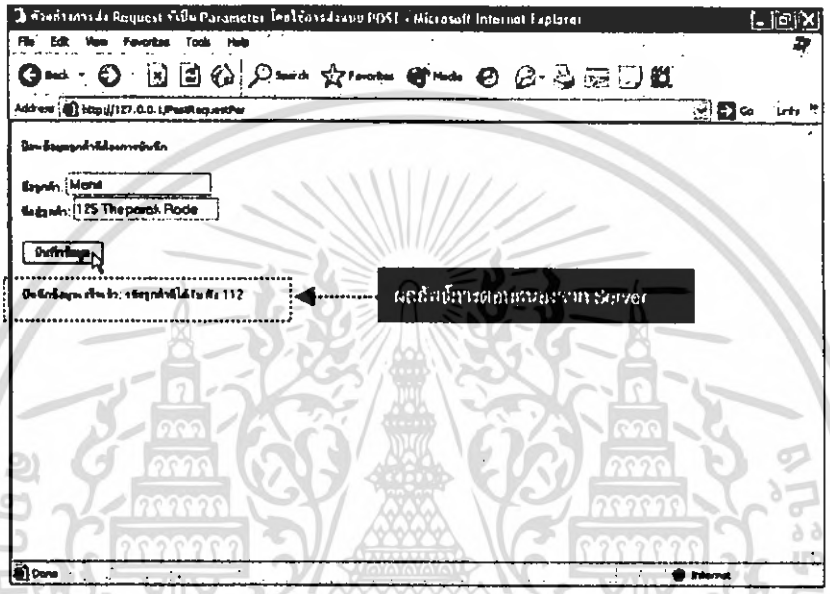
จากตัวอย่างที่ 2.18 ที่แสดงโค้ดของไฟล์ PostRequestPar.html การทำงานในโค้ดมี 3 ส่วน หลักๆที่สำคัญอย่างมากต่อการส่ง Request แบบ POST ได้แก่

บรรทัดที่ 17-22 เป็นฟังก์ชันสำหรับส่ง Request ไปยังฝั่ง Server

บรรทัดที่ 36-47 เป็นฟังก์ชั่นสำหรับนำข้อมูลจาก Form เข้ารหัสก่อนส่งไปยังฝั่ง Server

บรรทัดที่ 57-62 เป็นกลุ่มโค้ดสำหรับสร้าง Form เพื่อรับข้อมูลจาก User

คำสั่งในการส่ง Request แบบ POST ที่ประกอบด้วยโค้ดทั้ง 3 ส่วนที่กล่าวมา การทำงานเริ่มจากเมื่อ User ป้อนข้อมูลผ่าน Form ที่อยู่บนเว็บเพจ เสร็จแล้วคลิกปุ่ม "บันทึกข้อมูล" จากนั้น XMLHttpRequest จะส่งข้อมูลไปยังไฟล์ PostCustomerData.php ที่ทำงานอยู่ฝั่ง Server เมื่อฝั่ง Server รับ Request แล้วตอบสนองกลับมายัง Web Browser แล้ว XMLHttpRequest จะรับข้อมูลที่ตอบสนองกลับมาและส่งข้อมูลให้ JavaScript ทำหน้าที่ควบคุมการแสดงผล ดังรูปที่ 2.31



รูปที่ 2.31 แสดงเว็บเพจที่โต้ตอบกับ User ของไฟล์ PostRequestPar.html

จากรูปที่ 2.31 แสดงการโต้ตอบกับ User ของไฟล์ PostRequestPar.html เมื่อลูกค้าต้องการบันทึกข้อมูลสามารถป้อนข้อมูลและคลิกที่ปุ่ม "บันทึกข้อมูล" จากนั้น XMLHttpRequest Object จะเริ่มสร้าง Request และส่งไปยังฝั่ง Server ต่อไป

2.4 Java กับ JavaScript

โครงสร้างภาษาของ JavaScript มีความคล้ายคลึงกับ Java มาก โดย JavaScript เป็น คอมพลีเมนต์ (complement) ของ Java สามารถติดต่อกับส่วนต่าง ๆ ของจาวาแอปเพล็ตโดยสคริปต์ที่เขียนขึ้นมาได้ คำสั่งของ JavaScript สามารถนำมาใช้แสดง, กำหนดคุณสมบัติ, สอบถามสถานะ หรือ ควบคุมการกระทำของแอปเพล็ตและปลั๊กอิน นอกจากนี้ JavaScript ยังสนับสนุนรูปแบบนิพจน์และการควบคุมพื้นฐาน ของภาษา Java อีกด้วย JavaScript ได้ถูกออกแบบมาเพื่อใช้เป็นส่วนเพิ่มขยายในภาษา HTML โดยเฉพาะ ช่วยให้สามารถควบคุมเว็บเพจได้อย่างง่ายดาย เหมาะกับการทำงานอย่างรวดเร็ว และเน้นที่ความถูกต้องเป็นสำคัญ ภาษา Java ประกอบไปด้วย เอ็กซ์คลูซีฟ (exclusive) ของ class และ method ต้องมีการกำหนด class และ method และเน้น เรื่องความถูกต้อง โปรแกรมที่เขียนในภาษา Java จะมี

ความสมบูรณ์กว่าการเขียนด้วย Javascript JavaScript เป็นภาษาแบบอินเตอร์พรีเตอร์ (interpleter) ฉะนั้นเพียงเขียนคำสั่ง ในภาษา JavaScript เก็บไว้เป็น text file ร่วมกับเว็บ เพจ HTML ก็ทำงานได้แล้ว ไฟล์คำสั่งในภาษา JavaScript อาจมีส่วนขยายเป็น .htm หรือ .html เหมือนกับไฟล์เว็บเพจทั่วไป หรือมีส่วน ขยายเป็น .js ก็ได้ แต่ภาษา Java เป็นภาษาแบบ คอมไพเลอร์ (compiler) คำสั่งในภาษา Java จึงต้องเขียนเก็บไว้เป็น text file มีส่วนขยายเป็น .java หลังจากนั้นต้องนำไฟล์ดังกล่าวไปผ่านการคอมไพล์ให้เป็นไบต์โค้ด (ไฟล์ที่มีส่วนขยายเป็น class) เสียก่อน จากนั้นจึงนำไฟล์มาสร้าง เป็นอ็อบเจ็กต์และแอปเพล็ตเพื่อใช้งานต่อไป

Java คืออะไร

Java นอกจากเป็นชื่อกาแฟรสดีตามชื่อเกาะชวาของประเทศอินโดนีเซียแล้ว ที่เรามักจะเป็นตามหนังสือมักจะเป็นรูปกาแฟ ยังเป็นภาษาคอมพิวเตอร์สำหรับ ใช้งานในระบบเครือข่ายอินเทอร์เน็ตที่มีชื่อเสียงโด่งดังที่สุดเป็นภาษามาตรฐานระดับสูงที่มีความสามารถ ในการทำงานได้โดยไม่ยึดติดกับแพลตฟอร์มใด ๆ ของระบบคอมพิวเตอร์ ไม่ว่าจะเป็น เครื่องแบบพีซี , แมคอินทอช(Macintoch) , ซัน, Unix, Apple, เครื่องระดับมินิคอมพิวเตอร์จนถึงระดับซุเปอร์คอมพิวเตอร์ ลักษณะของภาษาจาวา จะมีความสามารถในการสร้างโปรแกรมขนาดเล็กที่เรียกว่า แอปเพล็ต Applets) สำหรับใช้งานในระบบ อินเตอร์เน็ตโดยทำงานร่วมกันกับโปรแกรมบราวเซอร์ มี Java Compiler เป็นตัวแปรภาษาซอร์ซโค้ด(Source Code) ให้กลายเป็นภาษา กลางที่เรียกว่า ไบต์โค้ด (Byte Code) ข้อดีของภาษาจาวาก็คือ โปรแกรมที่เขียนมีขนาดเล็ก สามารถนำไปประยุกต์ใช้งานได้ โดยตัวโปรแกรมจะอยู่บนเครื่องเซิร์ฟเวอร์ และ เมื่อใดที่มีการเรียกใช้งานจากเว็บบราวเซอร์ เซิร์ฟเวอร์ก็จะทำการส่งข้อมูลและ โปรแกรมที่ต้องการคือให้กับบราวเซอร์เพื่อไปทำการประมวลผลแสดงผลลัพธ์ในเว็บบราวเซอร์ต่อไป

JavaScript คืออะไร

JavaScript เป็นภาษายุคใหม่สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ตที่กำลังได้รับความนิยมอย่างสูง เราสามารถเขียน โปรแกรม JavaScript เพิ่มเข้าไปในเว็ปเพจเพื่อใช้ประโยชน์สำหรับงานด้านต่าง ๆ ทั้งการคำนวณ การแสดงผล การรับ-ส่งข้อมูล และที่ สำคัญคือ สามารถโต้ตอบกับผู้ใช้ได้อย่างทันทีทันใด นอกจากนี้ยังมีความสามารถด้านอื่น ๆ อีกหลายประการที่ช่วยสร้างความน่าสนใจให้ กับเว็บเพจของเราได้อย่างมาก ภาษาจาวาสคริปต์ถูกพัฒนาโดย เน็ตสเคปคอมมิวนิเคชันส์ (Netscape Communications Corporation) โดยใช้ชื่อว่า Live Script ออกมาพร้อมกับ Netscape Navigator2.0 เพื่อใช้สร้างเว็บเพจโดยติดต่อกับเซิร์ฟเวอร์แบบ Live Wire ต่อมาเน็ตสเคปจึงได้ร่วมมือกับ บริษัทซันไมโครซิสเต็มส์ปรับปรุงระบบของบราวเซอร์เพื่อให้สามารถติดต่อใช้งานกับภาษาจาวาได้ และได้ปรับปรุง LiveScript ใหม่เมื่อ ปี 2538 แล้วตั้งชื่อใหม่ว่า JavaScript

2.4.1 ลักษณะการทำงานของ JavaScript

JavaScript เป็นภาษาสคริปต์เชิงวัตถุ หรือเรียกว่า อ็อบเจ็กไอเรียลเต็ด (Object Oriented Programming) ที่มีเป้าหมายในการ ออกแบบและพัฒนาโปรแกรมในระบบอินเทอร์เน็ต สำหรับผู้เขียนเอาสารด้วยภาษา HTML สามารถทำงานข้ามแพลตฟอร์มได้ทำงานร่วมกับ ภาษา HTML และภาษาจาวาได้ทั้งทางฝั่งไคลเอนต์ (Client) และ ทางฝั่งเซิร์ฟเวอร์ (Server) โดยมีลักษณะการทำงานดังนี้ 1. Navigator JavaScript เป็น Client-Side JavaScript ซึ่งหมายถึง JavaScript ที่ถูกแปลทางฝั่งไคลเอนต์ (หมายถึงฝั่งเครื่อง คอมพิวเตอร์ของผู้ใช้ ไม่ว่าจะเป็นเครื่องพีซี เครื่องแมคอินทอช หรือ อื่น ๆ) จึงมีความเหมาะสมต่อการใช้งานของผู้ใช้ทั่วไปเป็นส่วนใหญ่ 2. LiveWire JavaScript เป็น Server-Side JavaScript ซึ่งหมายถึง JavaScript ที่ถูกแปลทางฝั่งเซิร์ฟเวอร์ (หมายถึงฝั่งเครื่อง คอมพิวเตอร์ของผู้ให้บริการเว็บ โดยอาจจะเป็นเครื่องของซัน ซิลิคอมกราฟิกส์ หรือ อื่น ๆ) สามารถใช้ได้เฉพาะกับ LIveWire ของเน็ตสเคปโดยตรง

ตารางที่2.5 จะแสดงต่อไป จะเปรียบเทียบผลิตภันฑ์ และ ภาษาที่ใช้ในแต่ละส่วน

โปรแกรม	ค่ายเนสเคป	ค่ายไมโครซอฟท์
Web Browser	Nescape Navigator	Internet Explorer
Client-Side Script	JavaScript	VBScript , JScript
Web Server	Nascape Enterprise Server	Internet Information Server
Server-Side Script	LiveWire	Active Server Page

ตารางที่2.5 เปรียบเทียบผลิตภันฑ์ และ ภาษาที่ใช้ในแต่ละส่วน

ภาษา Java Script เมื่อถูกนำไปพัฒนาต่อโดย Microsoft ได้เปลี่ยนชื่อเป็น JScript นอกจากนั้น Microsoft ยังได้พัฒนาภาษา VB Script ให้ใช้เขียนโปรแกรมบนเวบได้อีกด้วย ซึ่งในปัจจุบันโปรแกรมจากทางค่าย ไมโครซอฟท์จะได้รับความนิยมมากกว่า ดังนั้น JScript จึงกลายเป็นมาตรฐานหลักที่ใช้กันบน Internet

การแทรก JavaScript ใน HTML

การเขียน Java Script ลงในเอกสาร HTML ทำให้มันสามารถตัดสินใจ สามารถตอบโต้กับผู้ที่เข้าชมเวบเพจได้ ซึ่งมีวิธีการแทรกคำสั่ง JavaScript ลงใน HTML อยู่ 4 วิธีดังนี้

วิธีที่ 1 การใช้ tag <script>

รูปแบบ

```
<html>
<head>
<script language=JavaScript>
<!--
JavaScript Language...
```

```
//-->
</script>
</head>
</html>
```

เครื่องหมาย <!-- //--> เป็นเครื่องหมายที่ใช้เผื่อบางเครื่องใช้จาวาสคริปต์ไม่ได้
มันจะไม่แสดงอะไรออกมาเลยแม้โค้ดเราจะเขียนแล้วก็ตาม
การใช้งานในลักษณะนี้จะใช้แท็กคู่ <SCRIPT> และ </SCRIPT>
โดยจะวางที่ตำแหน่ง <HEAD> และ </HEAD> หรือ ที่ <BODY> และ </BODY> ก็ได้
แต่ตำแหน่ง HEAD จะเหมาะกับการประกาศตัวแปร และ การเขียน function ที่ยังไม่ได้ประมวลผล
ทันที แต่จะถูกเรียกใช้งานจาก Script ในส่วนที่อยู่ใน <BODY>

วิธีที่ 2 ใช้ใน Even Handler

ตัวอย่างภาษา HTML ที่มีการแทรก script เพื่อจัดการ event

```
<img src="xx.gif" name="pic">
< input type=button name=button1 onclick="pic.border++">
```

วิธีนี้เราจะใส่ลงในจาวาสคริปต์โค้ดเข้าไปในแท็ก HTML โดยตรง โดยใช้แอตทริบิวต์พิเศษที่เรียกว่า Even Handler เช่น onclick, onmouseover, onmouseout เป็นต้น

วิธีที่ 3 การใช้ ไฟล์ script

วิธีที่ 4 ใช้ JavaScript entity เป็นค่าแอตทริบิวต์ของแท็กใน HTML

2.4.2 โครงสร้างภาษา

Statements

เหมือนกับภาษาโปรแกรมทั่วไป ภาษา JavaScript จะถูกเขียนในแบบข้อความ ที่มีการจัดโครงสร้างเป็นประโยค(statement) และ กรอบ (block) ซึ่งแทนเซตของประโยค และ มีการทำหมายเหตุ (comment) ภายในประโยคคำสั่งคุณสามารถใช้งานตัวแปร เขียนนิพจน์การคำนวณ สร้างเงื่อนไข ต่างๆ ได้

Statements

ประโยคคำสั่งของ JavaScript ก็จะคล้ายกับประโยคในภาษาพูด ซึ่งจะใช้แทนความหมายของคำสั่งอย่างใดอย่างหนึ่ง เมื่อนำหลาย ๆ ประโยคคำสั่งมารวมกันเราก็จะเรียกว่าโปรแกรม ตัวอย่างเช่น

```
name = "Tommy";
greeting = "Welcome "+name;
```

โดยประโยคใน Java Script จะเขียนเป็น บรรทัด ปิดท้ายด้วย ; (semi-colon) ถ้าในหนึ่งบรรทัดอาจจะมีหลายคำสั่ง ให้คั่นแต่ละคำสั่งด้วย ; เช่น

```
name = "Tommy";greeting = "Welcome "+name
```

การเขียนโปรแกรมจะประกอบด้วยคำสั่งรวมกันเป็นชุด หรือ Block ซึ่งจะใช่เครื่องหมาย { } ในการกำหนดขอบเขตของ block

```
{ name = "Tommy";
  greeting = "Welcome "+name; }
```

เราจะใช้ block ในกรณีการสร้างเงื่อนไข หรือ การสร้างฟังก์ชัน

Comments

ใช้สำหรับสร้างหมายเหตุ เพื่อ อธิบายการทำงานของโปรแกรม ซึ่งคอมพิวเตอร์จะไม่สนใจแปลความหมายของคำที่อยู่ใน comment เราจึงสามารถเขียนอะไรก็ได้ไว้ใน comment จะไม่มีผลกับการทำงานของโปรแกรม สำหรับ comment ของ java script จะมี 2 แบบคือ

1. แบบ single-line เขียนหมายเหตุ 1 บรรทัด ใช้เครื่องหมาย // เช่น

   ```
   name="Tommy";  // ประกาศตัวแปร name
   ```

2. แบบ multiline ใช้เขียนหมายเหตุที่มีหลายบรรทัด ใช้เครื่องหมาย /* */ ครอบหมายเหตุไว้ เช่น

   ```
   /*
   ตัวอย่างโปรแกรม */
   name = "Tommy";
   ```

Expressions

นิพจน์ หรือ expression จะคล้ายกับ "คำ" ซึ่งมันจะต้องมีค่าในตัวเอง ตัวอย่าง

```
3.9                  // numeric literal
"Hello!"             // string literal
false                // boolean literal
null                 // literal null value
{x:1, y:2}           // Object literal
[1,2,3]              // Array literal
function(x){return x*x;}  // function literal
```

Operator

โอเปอร์เรเตอร์ที่ใช้ในจาวาสคริปต์จะคล้ายคลึงกับโอเปอร์เรเตอร์ในภาษา C , C++ หรือในภาษาจาวา ดังตัวอย่างดังตารางที่2.6ต่อไปนี้

โอเปอเรเตอร์ในการคำนวณ	ความหมาย
+	การบวก,การเชื่อมข้อความสตริง
-	การลบ
*	การคูณ

/	การหาร
%	การหารเอาเฉพาะเศษ
++	การเพิ่มข้อมูลเข้าไปในตัวแปรอีกหนึ่ง
--	การลดค่าข้อมูลตัวแปรลงไปอีกหนึ่ง

ตารางที่2.6โอเปอร์เรเตอร์การคำนวณในภาษา C , C++ หรือในภาษาจาวา

โอเปอเรเตอร์ทางตรรกศาสตร์	ความหมาย
== , !=	เท่ากับ, ไม่เท่ากับ
>, >=, =<, <	น้อยกว่า, น้อยกว่าหรือเท่ากับและอื่น ๆ ใช้เปรียบเทียบทางคณิตศาสตร์และเชิงสตริง
!	NOT
&&, ||	AND , OR
?	ตัดสินใจจากเงื่อนไข

ตารางที่2.7 โอเปอร์เรเตอร์ทางตรรกศาสตร์ในภาษา C , C++ หรือในภาษาจาวา

2.4.2 ชนิดข้อมูลและตัวแปร

ในภาษา JavaScript นั้นจะใช้ชนิดข้อมูลแบบ Dynamic นั้นคือไม่จำเป็นต้องประกาศตัวแปรก่อนการใช้งาน

ตัวแปรคืออะไร

ตัวแปร หรือ Variables จะถูกใช้สำหรับเก็บข้อมูล

การใช้งานตัวแปร

การประกาศตัวแปร

คุณสามารถประกาศตัวแปรได้โดยใช้ประโยคคำสั่ง var:

```
var strname = some value
```

คุณอาจจะประกาศตัวแปรโดยตัดคำว่า var ออกไปก็ได้:

```
strname = some value
```

การกำหนดค่าให้ตัวแปร

ใช้เครื่องหมาย = เช่น var name = "Tommy" **หมายถึงให้ตัวแปรชื่อว่า** name **เก็บค่าข้อความว่า "Tommy" สังเกตุว่าค่าของตัวแปรที่เป็นข้อความจะต้องอยู่ในเครื่องหมายคำพูด ถ้าไม่มีเครื่องหมายคำพูดจะหมายถึงชื่อตัวแปร**

การตั้งชื่อตัวแปร

ชื่อของตัวแปรมีความสำคัญมาก ในภาษา Java Script นั้นชื่อตัวแปรจะต้องเป็นภาษาอังกฤษ ตัวอักษรเล็กใหญ่จะถือว่าเป็นคนล่ะตัวแปร เช่น name กับ Name จะถือว่าเป็นคนละตัวแปร ข้อกำหนดทั่วไปของการตั้งชื่อ

- ต้องขึ้นต้นด้วยตัวอักษร a-z หรือ A-Z หรือ เครื่องหมาย _ เท่านั้น
- ตัวต่อไปสามารถเป็นได้ทั้ง ตัวอักษร และ ตัวเลข
- ห้ามมีช่องว่างระหว่างคำ
- ต้องไม่ตั้งชื่อซ้ำกับ คำสงวน

Reserved Words

break	delete	function	return	typeof
case	do	if	switch	var
catch	else	in	this	void
continue	false	instanceof	throw	while
debugger	finally	new	true	with
default	for	null	try	

ตารางที่ 2.8 Reserved Words

Future Reserved Words

abstract	double	goto	native	static
boolean	enum	implements	package	super
byte	export	import	private	synchronized
char	extends	int	protected	throws
class	final	interface	public	transient
const	float	long	short	volatile

ตารางที่ 2.9 Future Reserved Words

ชนิดข้อมูล Data Types

ใน Java Script มีชนิดข้อมูลแบบ primary data type 3 ชนิด มีแบบ composite data type 2 ชนิด และ มีข้อมูลชนิดพิเศษ 2 ชนิด

Primary (primitive) data types ได้แก่

- String
- Number

- Boolean

Composite (reference) data types ได้แก่

- Object
- Array

ชนิดข้อมูลแบบพิเศษได้แก่

- Null
- Undefined

ตัวแปรประเภท Object

ออบเจ็กต์ในจาวาสคริปต์ ประกอบด้วยกลุ่มสมาชิกที่เรียกว่า " คุณสมบัติ (Properties)" และกลุ่มของสมาชิกที่เป็นสิ่งที่ออบเจ็กต์สามารถทำงานได้เรียกว่า " เมธอด (method)" ซึ่งก็คล้าย ๆ กับฟังก์ชันในภาษา C

เพื่อจะให้มองเห็นภาพของออบเจ็กต์ง่าย ๆ จะยกตัวอย่างว่าสมมติเรามีออบเจ็กต์ชื่อ "appt" ที่ใช้ในการจัดการเกี่ยวกับการนัดหมาย (appoint) ออบเจ็กต์ของการนัดหมายจะมีคุณสมบัติ ดังนี้

- day คือวันที่นัดหมาย
- month คือเดือนที่มีการนัดหมาย
- time คือเวลาในการนัดหมาย
- who คือคนที่เรานัดหมาย
- why คือหัวข้อที่คุยกันในการนัดหมาย

แต่ละคุณสมบัติของออบเจ็กต์จะถูกอ้างอิง(ก็คือการเอาคุณสมบัติมาใช้นั่นแหละ) โดยการใช้จุด (.) เชื่อม เช่น appt.month appt.day เป็นต้น

การอ้างถึง Object ในเอกสาร

วัตถุต่าง ๆ ที่แสดงผลบนเวบ ก็เป็นข้อมูลแบบหนึ่งเช่นเดียวกัน ซึ่งข้อมูลเหล่านี้สามารถอ้างถึงได้โดยอาศัยชื่อที่ถูกต้อง ซึ่งวัตถุแต่ละสิ่งที่แสดงบนหน้าเวบนั้นจะมีชื่อตัวแปรของตน หรือ บางสิ่งอาจจะไม่มีชื่อ แต่อาจจะสามารถอ้างถึงได้โดยอาศัย การอ้างชื่อผ่านทางสิ่งอื่น เช่น

เราสามารถช่องป้อนข้อความ ตั้งชื่อว่า input1 ดังตัวอย่าง

```
<input type="text" name="input1" value="Test Data" style='color=red'>
```

เราก็จะสามารถเข้าถึงค่าตัวแปร input1 ได้ โดยไม่ต้องประกาศตัวแปรนี้ในภาษา javascript เนื่องจากช่องป้อนข้อความเป็น วัตถุแบบ Object ดังนั้นภายใน input1 จะมีค่าข้อมูลหลายอย่างอยู่ภายใน เช่น ค่าที่อยู่ในช่องข้อความ ค่าความกว้าง style สีพื้น สีตัวอักษร

<u>การใช้ Object ต่างๆ</u>

การใช้ String

ข้อความที่อยู่ในภาษา Java Script ทั้งหมดจะมีชนิดข้อมูลเป็นแบบ String ซึ่งใน String จะมีคำสั่งให้เราจัดการกับข้อความนั้นได้เช่น

```
s1 = "hello";
s2 = s1.bold();
```

จะได้ว่า s2 จะมีค่าเป็น <B>hello</B> โดยที่ s1 จะมีค่าเป็น hello เหมือนเดิม

*** การแสดงผลออกหน้าเวบจะใช้คำสั่ง document.write และ document.writeln

Properties

- length เป็นค่าความยาว หรือ จำนวนตัวอักษรใน string เช่น "hello".length จะมีค่า 5

Methods

String จะมี method จำนวนมาก คุณสามารถศึกษาการใช้งานโดยละเอียดจากคู่มือภาษา javaScript

Method	Description	NN	IE
anchor("anchorname")	Returns a string as an anchor	2	3
big()	Returns a string in big text	2	3
blink()	Returns a string blinking	2	
bold()	Returns a string in bold	2	3
charAt(index)	Returns the character at a specified position	2	3
charCodeAt(i)	Returns the Unicode of the character at a specified position	4	4
concat()	Returns two concatenated strings	4	4
fixed()	Returns a string as teletype	2	3
fontcolor()	Returns a string in a specified color	2	3
fontsize()	Returns a string in a specified size	2	3
fromCharCode()	Returns the character value of a Unicode	4	4
indexOf()	Returns the position of the first occurrence of a specified string inside another string. Returns -1 if it never occurs	2	3
italics()	Returns a string in italic	2	3
lastIndexOf()	Returns the position of the first occurrence of a specified string inside another string. Returns -1 if it never occurs. Note: This method starts from the right and moves left!	2	3

link()	Returns a string as a hyperlink	2	3
match()	Similar to indexOf and lastIndexOf, but this method returns the specified string, or "null", instead of a numeric value	4	4
replace()	Replaces some specified characters with some new specified characters	4	4
search()	Returns an integer if the string contains some specified characters, if not it returns -1	4	4
slice()	Returns a string containing a specified character index	4	4
small()	Returns a string as small text	2	3
split()	Splits a string into an array of strings	4	4
strike()	Returns a string strikethrough	2	3
sub()	Returns a string as subscript	2	3
substr()	Returns the specified characters. 14,7 returns 7 characters, from the 14th character (starts at 0)	4	4
substring()	Returns the specified characters. 7,14 returns all characters from the 7th up to but not including the 14th (starts at 0)	2	3
sup()	Returns a string as superscript	2	3
toLowerCase()	Converts a string to lower case	2	3
toUpperCase()	Converts a string to upper case	2	3

ตารางที่ 2.10 รายละเอียด Methods String

Date

ตัวแปรประเภท Date ใช้ในการคำนวณเกี่ยวกับวันที่ และ เวลา

การใช้ตัวแปร Date

เราใช้ตัวแปร Date ในการเก็บข้อมูลวันที่ และ เวลา

ตัวอย่างที่ 2.19 การสร้างตัวแปร : now = new Date(); จะได้ตัวแปร now เป็นวันเวลาปัจจุบัน

document.write(now);	Fri Jan 12 14:48:24 UTC+0700 2007
document.write(now.**getMonth()**);	0
document.write(now.**getYear()**);	2007

Math

การใช้คำสั่ง Math

Object Math ใช้สำหรับในการคำนวณฟังก์ชันทางคณิตศาสตร์ เช่น หารากที่สอง หาค่าตรีโกณมิติ cos,sin,tan เป็นต้น

Properties

Property	Description
E	Returns the base of a natural logarithm
LN2	Returns the natural logarithm of 2
LN10	Returns the natural logarithm of 10
LOG2E	Returns the base-2 logarithm of E
LOG10E	Returns the base-10 logarithm of E
PI	Returns PI
SQRT1_2	Returns 1 divided by the square root of 2
SQRT2	Returns the square root of 2

ตารางที่ 2.11 รายละเอียด Properties Math

Methods

Method	Description
abs(x)	Returns the absolute value of x
acos(x)	Returns the arccosine of x
asin(x)	Returns the arcsine of x
atan(x)	Returns the arctangent of x
atan2(y,x)	Returns the angle from the x axis to a point
ceil(x)	Returns the nearest integer greater than or equal to x
cos(x)	Returns the cosine of x
exp(x)	Returns the value of E raised to the power of x
floor(x)	Returns the nearest integer less than or equal to x
log(x)	Returns the natural log of x
max(x,y)	Returns the number with the highest value of x and y
min(x,y)	Returns the number with the lowest value of x and y

pow(x,y)	Returns the value of the number x raised to the power of y
random()	Returns a random number between 0 and 1
round(x)	Rounds x to the nearest integer
sin(x)	Returns the sine of x
sqrt(x)	Returns the square root of x
tan(x)	Returns the tangent of x

ตารางที่ 2.12 รายละเอียด Method Math

ประโยคเงื่อนไขและการควบคุม

การใช้คำสั่ง if

คำสั่ง if เป็นคำสั่งเพื่อสร้างทางเลือกให้กับโปรแกรม ทำให้โปรแกรมสามารถตัดสินใจได้ ตามเงื่อนไขที่กำหนด

รูปแบบคำสั่ง if (เงื่อนไข) { คำสั่งที่จะทำเมื่อเงื่อนไขเป็นจริง; } *else { คำสั่งที่จะทำเมื่อเป็นเท็จ }*

คำสั่งส่วนของ else ไม่จำเป็นต้องเขียนก็ได้

ความแตกต่างของการใช้ else กับ ไม่ใช้ คือ แบบใช้ else นั้น คอมพิวเตอร์จะตรวจสอบเงื่อนไขจากบรรทัดแรกก่อน ถ้าเงื่อนไขเป็นจริงก็ทำงาน โปรแกรมในปีกกา { } และจบการทำงาน เงื่อนไขที่อยู่ในส่วนของ else จะไม่ถูกตรวจสอบ

การวบลูปด้วย while

คำสั่ง while ช่วยให้เราสามารถสั่งให้คอมพิวเตอร์ ทำงานซ้ำชุดคำสั่ง เดิมได้หลายๆ ครั้ง จนกว่าจะออกจากเงื่อนไขที่กำหนด

เช่น ให้พิมพ์ข้อความว่า "รัก" เมื่อคำนวณคะแนนได้มากกว่า 60 คะแนน

มีรูปแบบดังนี้

```
while ( เงื่อนไข ) {
  ... ประโยคคำสั่งที่จะให้ทำงาน ซ้ำๆ เมื่อเงื่อนไขยังเป็นจริงอยู่
}
```

การวนลูปด้วย for

คำสั่ง for จะเป็นคำสั่งที่ใช้สำหรับการวนลูป คล้ายกับคำสั่ง while แต่จะออกแบบมาให้เหมาะกับการวนลูปตามจำนวนรอบ เพื่อลดข้อผิดพลาดของคำสั่ง while

รูปแบบการใช้ for(คำสั่งเริ่มต้น ; เงื่อนไข ; คำสั่งหลังทำงานแต่ละรอบ){ คำสั่งในลูป }

จากคำสั่ง while ที่ใช้ในการวนลูป

```
n=0;
while(n<3){
  document.writeln('X');
```

```
  n++;
}
```

สามารถเขียนให้อยู่ในรูปของ for ได้ดังนี้

```
for(n=0; n<3 ;n++){
  document.writeln('X');
}
```

จะเห็นว่าคำสั่ง for จะรวมเอา คำสั่งสำหรับการจัดการเงื่อนไขของการวนลูปทั้ง 3 ส่วน เอาไว้ในบรรทัดเดียวกัน ซึ่งจะทำให้ใช้งานได้สะดวกมากยิ่งขึ้น

รูปแบบการสร้างฟังก์ชัน

ฟังก์ชั่น คือ กลุ่มคำสั่งตั้งแต่ 1 คำสั่งขึ้นไปที่มาอยู่รวมกัน เพื่อทำงานใดงานหนึ่งโดยเฉพาะ ประโยชน์ของฟังก์ชั่นคือ ถ้ามีการเขียนโปรแกรมที่ต้องมีการทำงานใดงานหนึ่งที่ซ้ำกัน แทนที่จะเขียนคำสั่งนั้นหลาย ๆ ที เราก็เขียนกลุ่มคำสั่งหรือฟังก์ชั่นนั้นทีเดียว แล้วเวลาจะใช้งานก็เรียกฟังก์ชั่นที่เราเขียนเอา ไม่ต้องเขียนคำสั่งหลาย ๆ ที

รูปแบบของฟังก์ชั่น

```
function ชื่อฟังก์ชั่น ( พารามิเตอร์ ) {
   กลุ่มคำสั่งในฟังก์ชั่น
}
```

ฟังก์ชั่นทุกฟังก์ชั่นจะต้องมีการตั้งชื่อฟังก์ชั่นไม่ซ้ำกัน และต้องไม่เป็นคำสงวน เช่น ใช้คำว่า 'while', 'for' เป็นต้น ไม่ได้ นอกจากนี้การเรียกใช้งานฟังก์ชั่นสามารถเรียกโดยการอ้างถึงชื่อฟังก์ชั่นนั้นได้เลย และสามารถส่งผ่านข้อมูลใด ๆ ไปยังฟังก์ชั่นได้ โดยฟังก์ชั่นต้องมีการกำหนดพารามิเตอร์เป็นตัวรับข้อมูลที่ส่งมา

2.5 ประวัติความเป็นมาของภาษา PHP

PHP เป็นภาษาจำพวก scripting language คำสั่งต่างๆจะเก็บอยู่ในไฟล์ที่เรียกว่าสคริปต์ (script) และเวลาใช้งานต้องอาศัยตัวแปลชุดคำสั่ง ตัวอย่างของภาษาสคริปก็เช่น JavaScript, Perl เป็นต้น ลักษณะของ PHP ที่แตกต่างจากภาษาสคริปต์แบบอื่นๆ คือ PHP ได้รับการพัฒนาและออกแบบมา เพื่อใช้งานในการสร้างเอกสารแบบ HTML โดยสามารถ สอดแทรกหรือแก้ไขเนื้อหาได้โดยอัตโนมัติ ดังนั้นจึงกล่าวว่า PHP เป็นภาษาที่เรียกว่า server-side หรือ HTML-embedded scripting language เป็นเครื่องมือที่สำคัญชนิดหนึ่ง ที่ช่วยให้เราสามารถสร้างเอกสารแบบ Dynamic HTML ได้อย่างมีประสิทธิภาพและมีลูกเล่นมากขึ้น ถ้าใครรู้จัก Server Side Include (SSI) ก็จะสามารถเข้าใจการทำงานของ PHP ได้ไม่ยาก สมมุติว่า เราต้องการจะแสดงวันเวลาปัจจุบันที่ผู้เข้ามาเยี่ยมชมเว็บไซต์ในขณะนั้น ในตำแหน่ง ใดตำแหน่งหนึ่งภายในเอกสาร HTML ที่เราต้องการ อาจจะใช้คำสั่งในรูปแบบนี้ เช่น <!--#exec cgi="date.pl"--> ไว้ในเอกสาร HTML เมื่อ SSI ของ web server มาพบคำสั่งนี้ ก็จะกระทำคำสั่ง date.pl ซึ่งในกรณีนี้ เป็นสคริปต์ที่เขียนด้วยภาษา perl สำหรับอ่านเวลาจากเครื่องคอมพิวเตอร์ แล้วใส่ค่าเวลาเป็นเอาพุท (output) และแทนที่คำสั่งดังกล่าว ลงในเอกสาร HTML โดยอัตโนมัติ ก่อนที่จะส่งไปยังผู้อ่านอีกทีหนึ่ง อาจจะกล่าวได้ว่า PHP ได้รับการพัฒนาขึ้นมา เพื่อแทนที่ SSI รูปแบบเดิมๆ โดยให้มี

ความสามารถ และมีส่วนเชื่อมต่อกับเครื่องมือชนิดอื่นมากขึ้น เช่น ติดต่อกับคลังข้อมูลหรือ database เป็นต้น

PHP ได้รับการเผยแพร่เป็นครั้งแรกในปีค.ศ.1994 จากนั้นก็มีการพัฒนาต่อมาตามลำดับ เป็นเวอร์ชั่น 1 ในปี 1995 เวอร์ชั่น 2 (ตอนนั้นใช้ชื่อว่า PHP/FI) ในช่วงระหว่าง 1995-1997 และเวอร์ชั่น 3 ช่วง 1997 ถึง 1999 จนถึงเวอร์ชั่น 4 ในปัจจุบัน PHP เป็นผลงานที่เติบโตมาจากกลุ่มของนักพัฒนาในเชิงเปิดเผยรหัสต้นฉบับ หรือ OpenSource ดังนั้น PHP จึงมีการพัฒนาไปอย่างรวดเร็ว และแพร่หลาย โดยเฉพาะอย่างยิ่งเมื่อใช้ร่วมกับ Apache Webserver ระบบปฏิบัติอย่างเช่น Linux หรือ FreeBSD เป็นต้น ในปัจจุบัน PHP สามารถใช้ร่วมกับ Web Server หลายๆตัวบนระบบปฏิบัติการอย่างเช่น Windows 95/98/NT เป็นต้น

รายชื่อของนักพัฒนาภาษา PHP ที่เป็นแก่นสำคัญในปัจจุบันมีดังต่อไปนี้

- Zeev Suraski, Israel
- Andi Gutmans, Israel
- Shane Caraveo, Florida USA
- Stig Bakken, Norway
- Andrey Zmievski, Nebraska USA
- Sascha Schumann, Dortmund, Germany
- Thies C. Arntzen, Hamburg, Germany
- Jim Winstead, Los Angeles, USA
- Rasmus Lerdorf, North Carolina, USA

เนื่องจากว่า PHP ไม่ได้เป็นส่วนหนึ่งของตัว Web Server ดังนั้นถ้าจะใช้ PHP ก็จะต้องดูก่อนว่า Web server นั้นสามารถใช้สคริปต์ PHP ได้หรือไม่ ยกตัวอย่างเช่น PHP สามารถใช้ได้กับ Apache WebServer และ Personal Web Server (PWP) สำหรับระบบปฏิบัติการ Windows 95/98/NT ในกรณีของ Apache เราสามารถใช้ PHP ได้สองรูปแบบคือ ในลักษณะของ CGI และ Apache Module ความแตกต่างอยู่ตรงที่ว่า ถ้าใช้ PHP เป็นแบบโมดูล PHP จะเป็นส่วนหนึ่งของ Apache หรือเป็นส่วนขยายในการทำงานนั่นเอง ซึ่งจะทำงานได้เร็วกว่าแบบที่เป็น CGI เพราะว่า ถ้าเป็น CGI แล้ว ตัวแปลชุดคำสั่งของ PHP ถือว่าเป็นแค่โปรแกรมภายนอก ซึ่ง Apache จะต้องเรียกขึ้นมาทำงานทุกครั้งที่ต้องการใช้ PHP ดังนั้น ถ้ามองในเรื่องของประสิทธิ ภาพในการทำงาน การใช้ PHP แบบที่เป็นโมดูลหนึ่งของ Apache จะทำงานได้มีประสิทธิภาพมากกว่า

2.5.1 ลักษณะการทำงานของภาษา PHP

การสอดแทรกคำสั่งภาษา PHP ในเอกสาร HTM

เพื่อเป็นการบ่งบอกให้รู้ว่า ส่วนใดเป็นคำสั่ง PHP ที่อยู่ภายในเอกสาร HTML จึงได้มีการกำหนดสัญลักษณ์ไว้ดังนี้ ซึ่งสามารถทำได้หลายรูปแบบ เช่น

1. <? ... ?> (SGML style)
2. <?php ... ?> (XML style)
3. <script language="php"> ... </script> (JavaScript style)
4. <% ... %> (ASP style)

ที่นิยมก็คือแบบแรก โดยเริ่มต้นด้วย <? และจบด้วย ?> และตรงกลางจะเป็นคำสั่งในภาษา PHP เราสามารถวางคำสั่ง PHP ไว้ภายในเอกสาร HTML ตามที่ต้องการได้ อาจจะสลับกับ Tag ของภาษา HTML ก็ได้

คำสั่งแรกที่ง่ายที่สุดสำหรับการเรียนรู้ ก็คือคำสั่ง echo แล้วตามด้วยข้อความหรือสตริงค์ (string) ข้อความในภาษา PHP จะเริ่มต้นและจบด้วย double quote (") เหมือนในภาษาซี

การใช้ตัวแปรในภาษา PHP

สำหรับการเขียนโปรแกรมสำหรับภาษาคอมพิวเตอร์ระดับสูง สิ่งที่จะขาดเสียมิได้คือ การกำหนดและใช้ตัวแปร (variable) ตัวแปรในภาษา PHP จะเหมือนกับในภาษา Perl คือเริ่มต้นด้วยเครื่องหมาย dollar ($) โดยเราไม่จำเป็นต้องกำหนดแบบของข้อมูล (data type) อย่างเจาะจงเหมือนในภาษาซี เพราะว่า ตัวแปลภาษาจะจำแนกเองโดยอัตโนมัติว่า ตัวแปรดังกล่าว ใช้ข้อมูลแบบใด ในช่วงเวลานั้นๆ เช่น ข้อความ จำนวนเต็ม จำนวนที่มีเลขจุดทศนิยมตรรก เป็นต้น

สัญลักษณ์ \n หมายถึงการขึ้นบรรทัดใหม่ เป็น escape character ตัวหนึ่ง (สำหรับตัวอื่นๆ โปรดดูในตาราง) เมื่อพิมพ์ข้อความเป็นเอาพุต และโปรดสังเกตว่า สำหรับการใช้งานภายในเอกสาร HTML การขึ้นบรรทัดใหม่โดยใช้ \n จะแตกต่างจากการขึ้นบรรทัดโดยใช้
 ใน HTML

Escaped characters

\n	newline
\r	carriage
\t	horizontal tab
\\	backslash
\$	dollar sign

\"	double-quote
%%	percent

ตารางที่ 2.13 รายละเอียด Escaped characters

ตัวแปรตัวหนึ่ง อาจจะมีข้อมูลหลายแบบในช่วงเวลาที่ต่างกัน แต่การจะใช้งานบ้างครั้งจะต้องดูด้วยว่า เก็บไหร่จะใช้เป็นตัวเลขเท่านั้น และไม่ใช้คับข้อความเป็นต้น

ในกรณีนี้ เรากำหนดในตอนแรกว่า $x ให้เก็บค่า 10 ซึ่งเป็นจำนวนเต็ม ถ้าเรานำมาบวกกับ 15.5 ผลที่ได้ก็จะเป็น 25.5 ซึ่งกลายเป็นเลขทศนิยม แล้วเก็บไว้ในตัวแปร $y ต่อมากำหนดให้ตัวแปร $x เก็บสตริงที่เก็บข้อความ "abc" แล้วนำมาบวกกับ 15.5 ก็จะให้ผลที่ได้เป็นเลขทศนิยม เนื่องจากไม่สามารถนำข้อความมาบวกกับตัวเลขได้ แต่ PHP อนุญาตให้เราทำเช่นนั้นได้ในบางกรณี สมมุติว่าสตริงค์มีเฉพาะตัวเลขและสามารถเปลี่ยนเป็น เลขจำนวนเต็ม หรือจำนวนจริงได้โดยอัตโนมัติ เราก็นำสตริงค์ นี้มาบวกลบคูณหรือหารกับตัวแปรที่เก็บเป็นตัวเลขได้ ค่าคงที่สำหรับเลขจำนวนเต็ม อาจจะอยู่ในรูปของเลขฐานแปดหรือสิบหกก็ได้ ถ้าเป็นเลขฐานแปดจะมีเลขศูนย์นำ ถ้าเป็นเลขฐานสิบหกจะมี 0x นำหน้า

การอ่านและแปลงแบบข้อมูลในตัวแปรหรือค่าคงที่แบบเจาะจง

เราสามารถแปลงแบบข้อมูลจากแบบหนึ่งไปยังอีกแบบหนึ่ง (type casting) เช่น แปลงจากข้อความที่มีเฉพาะตัวเลขให้กลายเป็นเลขจำนวนเต็ม (int) หรือทศนิยม (double), (float), (real) หรือยังจะใช้คำสั่ง settype() ทำได้

การอ่านแบบข้อมูลของตัวแปรหรือค่าคงที่

ถ้าต้องการเช็คดูว่า ตัวแปรมีข้อมูลแบบใด เราสามารถใช้คำสั่ง gettype() ได้ ค่าที่ได้จากฟังก์ชันก็จะเป็น "integer" "double" หรือ "string" เป็นต้น

ตัวอย่างที่ 2.20

```
<?
echo gettype(0),"\n";
echo gettype(1.1),"\n";
echo gettype(""),"\n";
echo gettype((1 == 1)),"\n";
$var="abc";
if ( gettype($var)=="string" ) {
 echo "this is a string\n";
}
?>
```

เราอาจจะไม่ใช้ gettype() ก็ได้ แต่เลือกใช้ฟังก์ชัน is_long() สำหรับเช็คค่าที่เป็นเลขจำนวนเต็ม, is_string() สำหรับเช็คค่าที่เป็นสตริงค์, is_double() สำหรับค่าที่เป็นเลขทศนิยม, is_array() สำหรับค่าที่เป็นอาร์เรย์ หรือ is_object() สำหรับค่าที่เป็นอ็อบเจกต์ เป็นต้น ซึ่งจะให้ค่าแท่ากับ true (1) ถ้าตัวแปรมีแบบข้อมูล ตรงตามที่กำหนด

ตัวอย่างที่ 2.21

```
<?
unset($a);
$a="hello";
if (is_string($a) == true) {
 echo "\$a is a string <BR>\n";
}
unset($a);
$a[]="red";
$a[]="green";
$a[]="blue";
if (is_array($a) == true) {
 echo "\$a is an array of size ".count($a)."<BR>\n";
}
?>
```

สำหรับข้อความในภาษา PHP เราอาจจะใช้ single qoute แทน double quote ได้ แต่เวลาใช้งานร่วมกับ echo หรือ print() จะให้ผลต่างกัน ตัวแปลคำสั่งจะมองข้ามชื่อตัวแปรและรวมถึงพวก escape sequence ต่างๆด้วยที่อยู่ในข้อความที่ใช้ single quote

คำอธิบาย (หมายเหตุ) ในภาษา PHP

ถ้าเราต้องการเขียนคำอธิบายในส่วนใดๆก็ตามของสคริปต์ เราก็จะสามารถทำได้โดยใช้ /* ... */ เหมือนในภาษาซี หรือ // เหมือนในภาษาจาวา หรือ # เหมือน shell script โปรดสังเกตว่า // ใช้เขียนนำคำอธิบายในภายบรรทัดหนึ่งๆ เท่านั้น ส่วน # ใช้เริ่มต้นของบรรทัดที่เขียนคำอธิบาย

ตัวอย่างที่ 2.22

```
<?
# comment
$a = 41;  // set $a to 41.
$b =10;  // set $b to 10.
$b += $a; /* add $a to $b */
echo $b," \n";
?>
```

คำการใช้คำสั่งสำหรับคำนวณเลขคณิต

- บวก (+) เช่น $x + $y
- ลบ (-) เช่น $x - $y
- คูณ (*) เช่น $x * $y
- หาร (/) เช่น $x / $y
- หาเศษจากการหาร (%) หรือ โมดูลัส เช่น $x % $y การเศษจากการหาร โดยปรกติจะใช้กับเลขจำนวนเต็มเท่านั้น ถ้าใช้กับเลขมีจุดทศนิยม จะมีการปัดทิ้งเป็นจำนวนเต็มก่อน

กำหนดให้ $x มีค่าเท่ากับ 7 และ $y มีค่าเท่ากับ 4

ตัวอย่างที่ 2.23

$x + $y **11**

$x - $y	3
$x * $y	28
$x / $y	1.75
$x % $y	3

กำหนดให้ $x มีค่าเท่ากับ 2.5 และ $y มีค่าเท่ากับ 4

ตัวอย่างที่ 2.24

$x + $y	6.5
$x - $y	-1.5
$x * $y	1.0
$x / $y	0.615
$x % $y	2

การเพิ่มหรือลดค่าของตัวเลขในตัวแปรทีละหนึ่งตามแบบภาษาซีหรือจาวา

- $x++ เพิ่มค่าขึ้นอีกหนึ่ง
- ++$x เพิ่มค่าขึ้นอีกหนึ่ง
- $x-- ลดค่าลงอีกหนึ่ง
- --$x ลดค่าลงอีกหนึ่ง ความแตกต่างของการวาง ++ หรือ -- ไว้ข้างหน้าหรือข้างหลัง คือดูว่า จะอ่านค่าของตัวแปรก่อน (ในกรณีที่มีการอ่านค่าของตัวแปร) หรืออ่านค่าหลังจากการเพิ่มหรือลด

การกำหนดค่าของตัวแปรที่เป็นตัวเลขหรือสตริงค์โดยใช้ assignment operators

การกำหนดค่า (assignment) หรือเปลี่ยนแปลงค่าให้แก่ตัวแปร จะใช้โอเปอร์เรเตอร์ (assignment operators) ได้ในหลายๆรูปแบบ เหมือนอย่างที่ใช้ในภาษาซี

การใช้ตัวแปรเป็นชื่อของตัวแปร

ภาษา PHP เปิดโอกาสให้เราสามารถเลือกหรือเปลี่ยนชื่อของตัวแปรได้ ตัวอย่างที่ 2.25เช่น

```
<?
$a = "var1";
$$a = 10.3;
echo "$a ${$a} $$a <BR>\n";
echo "$var1 <BR>\n";
?>
```

จากตัวอย่างข้างบน เรากำหนดให้ตัวแปร $a เก็บสตริงค์ "var1" และจะใช้เป็นชื่อของตัวแปรอีกตัวหนึ่ง โดยทางอ้อม $$a เป็นการอ้างถึงตัวแปรที่มีชื่อเดียวกับค่าของตัวแปร $a (ในกรณีนี้คือ var1) ดังนั้นถ้า

เราเขียนว่า $$a หรือ $var1 ก็หมายถึงตัวแปรตัวเดียวกัน ถ้าต้องการแสดงค่าของ $$a โดยใช้คำสั่ง echo โดยอยู่ในสตริงค์ (ระหว่าง double quotations) เราจะต้องเขียน ${$a} ไม่ใช่ $$a เพราะว่า ถ้าเขียนตามแบบหลัง ตัวแปลคำสั่งจะอ่านค่า $a ก่อนแล้วแทนที่ลงในข้อความ ซึ่งจะได้ $var1แทนที่จะเป็นการอ่านค่าของ $var1

เทคนิคนี้ยังสามารถใช้ได้กับฟังก์ชัน ตัวอย่างที่ 2.26เช่น

```
<?
function foobar() {
  echo "foobar<BR>\n";
}
function callFunc ($f) {
  if ( is_string($f) == true) {
    $f();
  }
}
callFunc("foobar");
?>
```

ตัวอย่างที่2.26ข้างบนอาจจะทำให้เกิดปัญหาถ้าสมมุติว่า $fเป็นชื่อของฟังก์ชันที่ไม่มีอยู่จริง วิธีตรวจสอบคือ การใช้ฟังก์ชัน function_exists() ดังต่อไปนี้

ตัวอย่างที่ 2.27

```
<?
function MyFunc() {
 print ("ok..<BR>\n");
}
$f="myFunc";
if ( function_exists($f) ) {
  $f();
}
else {
  echo "$f does not exist!";
}
?>
```

การกำหนดค่าคงที่

ในภาษา PHP มีการทำสัญลักษณ์ให้เก็บค่าคงที่ เช่น อาจจะเป็นสตริงค์หรือตัวเลขก็ได้ สามารถทำได้โดยใช้ คำสั่ง DEFINE() สัญลักษณ์ที่กำหนดโดยคำสั่ง DEFINE() จะเหมือนกันตัวแปรทั่วๆไป แต่แตกต่างตรงที่ว่า เมื่อนิยามแล้วจะเปลี่ยนแปลงค่าอีกไม่ได้

ตัวอย่างที่ 2.28

```
<?
define(PI, 3.141592654);
define(YES, true);
define(NO, false);
define("AUTHOR", "RWS");
echo (PI/3),"<BR>\n";
echo "AUTHOR=".AUTHOR."<BR>\n";
echo "YES=".YES."<BR>\n";
?>
```

นอกจากสัญลักษณ์ที่ผู้ใช้นิยามขึ้นมาได้เองแล้วยังมีสัญลักษณ์กลุ่มหนึ่งที่ได้มีการนิยามไว้ก่อนแล้วในภาษา PHP ตัวอย่างที่ 2.29เช่น

__FILE__	เก็บชื่อของไฟล์สคริปต์
__LINE__	เก็บเลขบรรทัดภายในสคริปต์ในตอนที่ใช้
TRUE	มีค่าเป็นจริง
FALSE	มีค่าเป็นเท็จ
PHP_VERSION	เก็บเวอร์ชั่นของ PHP
PHP_OS	เก็บชื่อระบบปฏิบัติการที่ใช้ เช่น Linux

การทำขั้นตอนซ้ำหรือวนลูป

การวนลูปหรือสร้างลูปเพื่อทำงานซ้ำเป็นส่วนประกอบสำคัญของโปรแกรมคอมพิวเตอร์ ในภาษา PHP ก็จะใช้โครงสร้างเหมือนภาษาซี ดังต่อไปนี้

- while-do loop
- do-while loop
- for-loop

ตัวอย่างที่ 2.30การใช้ while-do loop เพื่อคำนวณค่า เลขยกกำลังสอง ซึ่งมีเลขฐานตั้งแต่ 1 ถึง 10

```
<?
$x = 1;
while ($x <= 10) {
 echo $x*$x,"\n";
 $x++;
}
?>
```

เริ่มต้นด้วยการกำหนดตัวแปร $x ให้มีค่าเป็นหนึ่ง ซึ่งในกรณีนี้ เราใช้เป็นเลขฐาน ในการคำนวณเลขยกกำลังสอง เมื่อเข้าสู่การวนลูปแบบ while-do จะมีการตรวจดูเงื่อนไข ของการวนลูปในแต่ละครั้งว่าเงื่อนไขเป็นจริงอยู่หรือไม่ ในกรณีนี้ เรากำหนดเงื่อนไขในการวนลูปไว้ว่า ถ้าค่าของ $x มีค่าน้อยกว่าหรือเท่ากับ 10 ก็ให้ทำคำสั่งที่อยู่ภายในลูป ซึ่งก็คือ echo $x*$x,"\n"; โดยจะพิมพ์ค่าของผลคูณซึ่งหมายถึงเลขยกกำลังสองนั่นเอง หลังจากนั้น ก็ให้เพิ่มค่าของ $x ทีละหนึ่งในการวนลูปแต่ละครั้ง ค่าของ $x จะเพิ่มขึ้นเรื่อยๆจนมีค่ามากกว่า 10 เมื่อถึงเวลานั้น ก็จะเป็นการจบการวนลูป เพราะว่าเราจะได้ว่า เงื่อนไข ($x <= 10) มีค่าเป็นเท็จ

สมมุติว่า ถ้าเปลี่ยนจาก $x++ เป็น $x-- ปัญหาก็จะเกิดตามมาเวลาใช้งาน คือ แทนที่จะวนลูปแค่สิบครั้ง ก็กลับกลายเป็นว่า เป็นการวนลูปนับครั้งไม่ถ้วน เพราะว่า ค่าของ $x จะลดลงเรื่อยๆในการวนลูปแต่ละ

ครั้ง คือเป็นลบ และค่าเป็นลบจะน้อยกว่า 10 เสมอ (ยกเว้นแต่ว่า เมื่อถึงจุดเวลาหนึ่งค่าเป็นลบมากๆ จะกระโดดกลับเป็นบวก)

ตัวอย่างที่ 2.31 การใช้ do-while loop เพื่อคำนวณค่าเลขยกกำลังสอง ซึ่งมีเลขฐานตั้งแต่ 1 ถึง 10

```
<?
$x = 1;
do {
 echo $x*$x,"<BR>\n";
 $x++;
} while ($x < 10);
?>
```

โปรดสังเกตความแตกต่างระหว่างการใช้ while-do และ do-while โดยเฉพาะตรงเงื่อนไข ในการจบการวนลูป ในกรณีของ do-while เราจะกระทำขั้นตอนในลูปก่อนหนึ่งครั้ง แล้วค่อยตรวจดูว่า เงื่อนไขในการวนลูปเป็นจริงหรือไม่ ความแตกต่างนี้ เราสามารถจำได้ง่ายๆ คือว่า ถ้าใช้ do-while จะต้องมีการทำคำสั่ง ภายในลูปหนึ่งครั้งเสมอ แม้ว่าเงื่อนไขโดยเริ่มต้นจะเป็นเท็จก็ตาม ซึ่งแตกต่างจาก while-do ถ้าเงื่อนไขเป็นเท็จตั้งแต่เริ่ม ก็จะไม่มีการทำคำสั่งที่อยู่ในลูป

อีกแบบหนึ่งสำหรับการวนลูปคือใช้ for-loop ทำได้ตามตัวอย่างที่ 2.32 ต่อไปนี้

```
<?
for ($x = 1; $x <=10; $x++) {
 echo $x*$x,"<BR>\n";
}
?>
```

ในบรรทัดที่เริ่มต้นด้วย for ระหว่างวงเล็บเปิดและปิด จะถูกแบ่งเป็นสามส่วนโดยเครื่องหมาย semicolon (;) ในส่วนแรกเราสามารถใส่คำสั่งที่ต้องการจะกระทำก่อนเข้าลูป ส่วนแรกนี้จะมีหรือไม่มีก็ได้ ในส่วนที่สองจะเป็นเงื่อนไขสำหรับการทำ loop และในส่วนที่สามจะคำสั่งที่จะต้องทำเป็นการจบท้ายลูปในแต่ละครั้ง หลักการทำงานของ for-loop จะคล้ายกับ while-do-loop

การใช้งาน for-loop และวางตำแหน่งส่วนต่างๆ อาจจะไม่จำเป็นต้องทำเหมือนกันแต่ให้ผลเหมือนกัน ตัวอย่างที่ 2.33 เช่น

```
<?
$x=1;
for ( ; $x <=10; $x++) {
  echo $x*$x,"<BR>\n";
}
$x=1;
for ( ; $x <=10; ) {
  echo $x*$x,"<BR>\n";
  $x++;
}
?>
```

จากตัวอย่างที่ 2.33 ข้างบนที่ผ่านๆมา เป็นการวนลูปจะใช้การนับเลขเพิ่มขึ้นทีละหนึ่ง เรายังสามารถเขียนใหม่โดยเป็นการนับเลขลดลง ยกตัวอย่างเช่น เราต้องการจะพิมพ์ตัวเลขเรียงลำดับจาก 10,9,8...,1 ก็อาจจะเขียนคำสั่งได้ดังตัวอย่างที่ 3.34 นี้

```
<?
for ($x=10 ; $x >0; $x--) {
  echo $x,"<BR>\n";
}
?>
```

การใช้งาน for-loop ก็จะเหมือนกับเวลาใช้ในภาษาซี ในหลายๆเรื่อง เช่น เราสามารถใส่คำสั่งได้มากกว่าหนึ่งโดยใช้เครื่องหมาย (,) เป็นตัวแยก ตัวอย่างที่ 3.35 เช่น

```
<?
for ($x=1, $y=0 ; $x < 10; $x++, $y--) {
  echo "$x $y<BR>\n";
}
?>
```

การแบ่งสายงานโดยจำแนกตามเงื่อนไขแบบ if-else

ในบางครั้งมีความจำเป็นต้องจำแนกเงื่อนไขในการทำงาน โดยแต่ละเงื่อนไขจะกำหนดกรณี เพื่อทำคำสั่งหรือกลุ่มของคำสั่ง ซึ่งอาจจะแตกต่างจากคำสั่งในกรณีอื่น ในภาษา PHP จะใช้ โครงสร้าง if หรือ if-else ในการจำแนกกรณีตามเงื่อนไข

ตัวอย่างที่ 2.36

```
<?
if ($x == 0)
 echo $x," is zero<BR>\n";
else if ($x > 0)
 echo $x," is positive<BR>\n";
else
 echo $x," is negative<BR>\n";
?>
```

จากตัวอย่าง ถ้า $x มีค่าเป็นศูนย์ตามเงื่อนไข ก็จะทำคำสั่ง echo $x," is zero
\n"; ถ้าเงื่อนไขแรกเป็นเท็จ ก็จะเงื่อนไขที่สองว่า $x มีค่ามากกว่าศูนย์หรือไม่ ถ้าใช่ ก็ทำคำสั่ง echo $x," is positive
\n"; ถ้าเงื่อนที่สองเป็นเท็จอีก ก็ให้ทำคำสั่งในกรณีสุดท้ายคือ $x จะต้องมีค่าเป็นลบ ถ้าในแต่ละกรณีต้องมีการทำคำสั่งมากกว่าหนึ่ง คือ เป็นกลุ่มคำสั่ง จะต้องใช้ { } มากำหนดขอบเขต (scope) เช่นตัวอย่างที่ 2.37

```
<?
if ($x == 0) {
 echo $x;
 echo " is zero.<BR>\n";
}
else if ($x > 0) {
 echo $x;
 echo " is positive.<BR>\n";
}
else {
 echo $x;
 echo " is negative.<BR>\n";
```

```
}
?>
```

โปรดสังเกตว่า { } ไม่ต้องมีเครื่องหมาย ; ต่อท้าย

ในภาษา PHP มีการกำหนด elseif (เงื่อนไข) ขึ้นมาใช้ ซึ่งไม่มีอะไรแตกต่างจาก else if (เงื่อนไข) โครงสร้างแบบ (เงื่อนไข) ? นิพจน์ : นิพจน์ แบบที่ใช้กันในภาษาซีนั้น ก็ใช้ได้เช่นกัน ตัวอย่างที่ 2.38 เช่น

```
<?
$x= -0.1035;
echo (($x < 0) ? -$x : $x),"<BR>\n";
?>
```

การใช้ break และ continue ภายในลูป

คำสั่ง break และ continue ภายในลูปอย่างที่ใช้กันในภาษาซี ก็นำมาใช้กับภาษา PHP ได้ ตัวอย่างที่ 2.39 เช่น

```
<?
unset($a);
$a[]=1;
$a[]=2;
$a[]=3;
$a[]="red";
$a[]="green";
$a[]="blue";
$a[]="none";
$i=0;
$found="not found";
for ($i=0; $i < count($a); $i++) {
 if ( is_long($a[$i]) ) { // skip all integer elements
   continue;
 }
 if ($a[$i] == "blue") {
   $found=$a[$i];
   break;
 }
}
echo $found,"<BR>\n";
?>
```

คำสั่ง continue บังคับให้ไปเริ่มต้นทำขั้นตอนในการวนลูปครั้งต่อไป ส่วน break นั้นส่งผลให้หยุดการทำงานของลูป

การแบ่งสายงานโดยจำแนกตามเงื่อนไขแบบ switch-case

นอกเหนือจากการใช้ if-else ในการจำแนกกรณีตามเงื่อนไขแล้ว เรายังสามารถใช้โครงสร้างแบบ switch-case ได้ ตัวอย่างที่ 2.40 เช่น

```
switch ($day) {
 case 1 :
   echo "Monday<BR>\n";
   break;
```

```
case 2 :
 echo "Tuesday<BR>\n";
 break;
case 3 :
 echo "Wednesday<BR>\n";
 break;
case 4 :
 echo "Thurday<BR>\n";
 break;
case 5 :
 echo "Friday<BR>\n";
 break;
case 6:
 echo "Saturday<BR>\n";
 break;
case 7 :
 echo "Sunday<BR>\n";
 break;
default :
 echo "error<BR>\n";
}
```

ถ้าตัวแปร $day มีค่าที่อยู่ระหว่าง 1 ถึง 7 ก็จะพิมพ์ชื่อวันเป็นภาษาอังกฤษ ถ้าตัวแปรมีค่านอกเหนือจากนั้น ซึ่งในกรณีจะเป็น default ในโครงสร้างแบบ switch-case ก็จะพิมพ์คำว่า error เพื่อให้ผู้ใช้ทราบ โปรดสังเกตว่า ในแต่ละกรณี จะต้องจบด้วยคำสั่ง break; ยกเว้นแต่ของ default ซึ่งจะมีหรือไม่ก็ได้ ถ้าเราไม่ได้ใส่คำสั่ง break; เอาไว้ โปรแกรมก็จะกระทำคำสั่งทุกคำสั่งในกรณีที่อยู่ถัดมา

การจำแนกกรณีไม่จำเป็นต้องอาศัยเฉพาะตัวแปรที่เก็บค่าจำนวนเต็มเท่านั้น ข้อมูลแบบอื่นก็ใช้ได้ เช่น ใช้ข้อความเป็นตัวจำแนกกรณี เช่นตัวอย่างที่ 2.41

```
switch ($answer) {
 case "yes" :
   echo "The user said 'yes'.\n";
   break;
 case "no" :
   echo "The user said 'no'.\n";
   break;
 default:
   echo "The user said neither 'yes' nor 'no'.\n";
}
```

โปรดสังเกตว่า การจำแนกโดยใช้ข้อความนี้ จะดูความแตกต่างระหว่างตัวพิมพ์เล็กหรือใหญ่ด้วย ในบางครั้งเราอาจจะไม่จำเป็นต้องใส่ break; ก็ได้ ตัวอย่างที่ 2.42 เช่น

```
switch ($answer) {
 case "yes" :
 case "no" :
   echo "The user said '",$answer,"'.\n";
   break;
 default:
   echo "The user said neither 'yes' nor 'no'.\n";
}
```

ตารางการเปรียบเทียบตัวเลขสำหรับสร้างเงื่อนไข

==	เท่ากับ
>	มากกว่า
>=	มากกว่าหรือเท่ากับ
<	น้อยกว่า
<=	น้อยกว่าหรือเท่ากับ
!=	ไม่เท่ากับ

ตารางที่ 2.14 ตารางการเปรียบเทียบตัวเลขสำหรับสร้างเงื่อนไข

เราสามารถสร้างเงื่อนไขจากการเปรียบเทียบมากกว่าน้อยกว่านี้ได้ซับซ้อนมากขึ้นโดยใช้ "และ" "หรือ" "ไม่" มาประกอบ ตัวอย่างที่ 2.43เช่น

($x == -1) \|\| ($x==1)	ถ้า $x มีค่าเท่ากับ -1 หรือ 1 จะได้เงื่อนไขเป็นจริง นอกเหนือจากนั้นเป็นเท็จ
($x < 10) && ($x >1)	ถ้า $x มีค่าน้อยกว่า 10 และ มากกว่า 1 ก็จะได้เงื่อนไขที่เป็นจริง นอกเหนือจากนั้นเป็นเท็จ
! ($x==0)	ถ้า $x ไม่เท่ากับศูนย์ ก็ได้เงื่อนไขเป็นจริง นอกเหนือจากนั้นเป็นเท็จ

การใช้ || และ && มีลักษณะการทำงานเหมือนในภาษาซี อย่างกรณีของ ($x || $y) ถ้า $x เป็นจริงจะไม่มีการพิจารณา $y และสำหรับ ($x && $y) ถ้า $x เป็นเท็จแล้วจะไม่มีการพิจารณา $y ต่อ

การคำนวณเลขคณิตในระดับบิต

การคำนวณแบบบิตที่ใช้ในภาษาซี ก็ใช้ได้กับภาษา PHP ตามตารางที่ 2.15ข้างล่างนี้

$x & $y	AND
$x | $y	OR
$x ^ $y	XOR
~ $x	NOT
$x << $y	SHIFT LEFT
$x >> $y	SHIFT RIGHT

ตารางที่ 2.15 การคำนวณแบบบิต

การใช้อาร์เรย์ (Array)

อาร์เรย์ในภาษา PHP นั้นจะแตกต่างจากอาร์เรย์ในภาษาซีหรือจาวาตรงที่ว่า อาร์เรย์ในภาษา PHP มีขนาดที่เปลี่ยนแปลงได้ หรือจะเรียกว่า dynamic array หรือ vector (สำหรับอาร์เรย์มิติเดียว) เริ่มต้นอาจจะแจ้งใช้ตัวแปรแบบอาร์เรย์ พร้อมเจาะจงขนาดเริ่มแรก เช่น มีขนาดเป็นศูนย์ก็ได้

ตัวอย่างที่ 2.44

```
$myarray[]=3;
$myarray[]=1.1;
$myarray[]="abc";
```

แต่เมื่อใช้อาร์เรย์ไป ขนาดของมันจะปรับเปลี่ยนได้ คือขยายจำนวนข้อมูลที่เก็บอยู่ภายในอาร์เรย์ ตามจำนวนข้อมูลที่เราใส่เพิ่มเข้าไป จากตัวอย่างข้างบน ในกรณีที่เรามิได้กำหนดเลขดัชนี (index) ก็หมายความว่า จะมีการขยายขนาดของอาร์เรย์เพิ่มขึ้นอีกหนึ่งโดยอัตโนมัติ ทุกครั้งที่เราใส่ข้อมูลที่อยู่ทางขวา และค่าที่เรากำหนดจากทางขวามือ และจะเก็บไว้ในที่ใหม่ของอาร์เรย์ เราไม่ต้องคำนึงถึงเรื่องการจอง หรือ ปลดปล่อยหน่วยความจำของอาร์เรย์ เหมือนอย่างในกรณีของอาร์เรย์ แบบไดนามิกในภาษาซี

นอกจากนั้นข้อมูลแต่ละตัวในอาร์เรย์ไม่จำเป็น ต้องเป็นข้อมูลชนิดเดียวกัน เช่น อาจจะมีทั้งจำนวนเต็ม เลขทศนิยม และข้อความ ปะปนกันไป ตัวอย่างที่ 2.45 เช่น

```
<?
$myarray[0] = 1;
echo "number of elements =".count($myarray)."<BR>\n";
$myarray[1] = "abc";
echo "number of elements =".count($myarray)."<BR>\n";
$myarray[2] = 1.3;
echo "number of elements =".count($myarray)."<BR>\n";
$myarray[]= 13+10;  // the same as $myarray[3]= 13+10;
echo "number of elements =".count($myarray)."<BR>\n";
for ($i=0; $i < 4; $i++) {
   echo $myarray[$i]." \n";
}
?>
```

ถ้าเราต้องการจะทราบจำนวนของข้อมูลที่มีอยู่ในอาร์เรย์เราจะใช้คำสั่ง count()

เทคนิคหนึ่งที่ใช้ในการสร้างอาร์เรย์ที่เก็บหลายๆข้อความหรือสตริงค์ คือ แทนที่เราจะกำหนดค่าของสมาชิก ในอาร์เรย์ทีละตัว เราจะสร้างได้โดยอัตโนมัติ โดยเก็บสตริงค์เหล่านั้นไว้ในสตริงค์เพียงอันเดียวโดยมีสัญลักษณ์ | เป็นตัวแยก และก็แล้วใช้ฟังก์ชันเป็นตัวแบ่งเพื่อสร้างอาร์เรย์อีกที

ตามตัวอย่างที่ 2.46

```
<?
// create empty array
$a=array();
// define string containing color names separated by | (pipe)
$color_names="red|green|blue";
```

```
// create array from string
$a=explode("|",$color_names);
while ( $color=each($a) ) {
 echo "$color[1]<BR>\n"; // note: $color[0] contains the index (0,1,2...)
}
?>
```

ลองดูอีกตัวอย่างที่2.47 ใช้ฟังก์ชัน explode() สร้างอาร์เรย์โดยอัตโนมัติสำหรับใส่ไว้ใน FORM ในส่วนของ SELECT เป็นเมนูให้เลือก

```
<?
// create selection list from a given string
function str2select($str, $delim) {
 $options = explode($delim,$str);
 $num = count($options);
 for( $i=0; $i < $num;$i++) {
  echo "<option> $options[$i]</option>\n";
 }
}
$select_str="10 บาท|20 บาท|30 บาท|40 บาท|50 บาท|100 บาท|200 บาท|500
บาท|1000 บาท";
?>
<FORM>
<SELECT NAME="testform">
<? str2select($select_str,"|"); ?>
</SELECT>
</FORM>
```

การใช้อาร์เรย์สองมิติ (2 Dimension Array)

ถ้าเราต้องการจะใช้อาร์เรย์แบบสองมิติ (หรือมากกว่า) ก็ทำได้เช่นกัน คือชื่อตัวแปรแล้วตามด้วย [..][..] ตัวอย่างที่ 2.48 เช่น

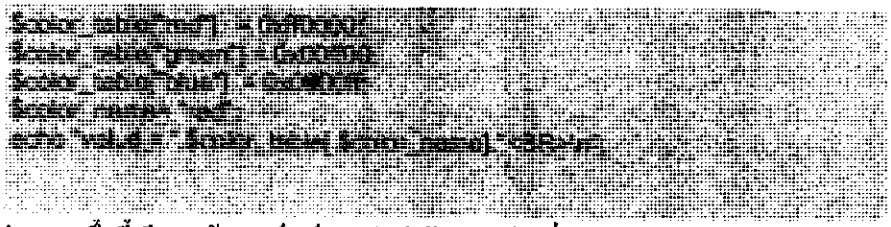
```
<?
$dim = 3;
for ($row=0; $row <= $dim; $row++) {
 for ($column=0; $column <= $dim; $column++) {
  $myarray2[$row][$column] = 4*$row + $column;
  echo $myarray2[$row][$column],".";
 }
 echo "<BR>\n";
}
?>
```

สังเกตว่า สำหรับการใช้งานตัวแปรที่เป็นอาร์เรย์ เราไม่จำเป็นต้องแจ้งใช้ตัวแปรที่เป็นอาร์เรย์ พร้อมกำหนดขนาดก่อนการใช้งาน

อาร์เรย์แบบเชื่อมโยงหรือ associative array

การเก็บข้อมูลในอาร์เรย์แบบนี้จะใช้กับข้อมูลที่จัดเก็บเป็นคู่ๆไป ซึ่งแตกต่างจากอาร์เรย์แบบแรกที่เราได้ทำความรู้จัก ตัวอย่างเช่น ใช้ทำ lookup table เช่น สมมุติว่า "red" ให้แทนค่า 0xff0000 "green" ให้แทนค่า 0x00ff00 และ "blue" 0x0000ff โดยเก็บไว้ในอาร์เรย์ชื่อ $color_table ตามตัวอย่างที่ 2.49

ต่อไปนี้



หรืออีกรูปแบบหนึ่งที่เขียนสร้างอาร์เรย์ดังกล่าวได้ โดยใช้คำสั่ง array()

ตัวอย่างที่ 2.50

```
$color_table = array(
   "red"   => 0xff0000,
   "green" => 0x00ff00,
   "blue"  => 0x0000ff
);
```

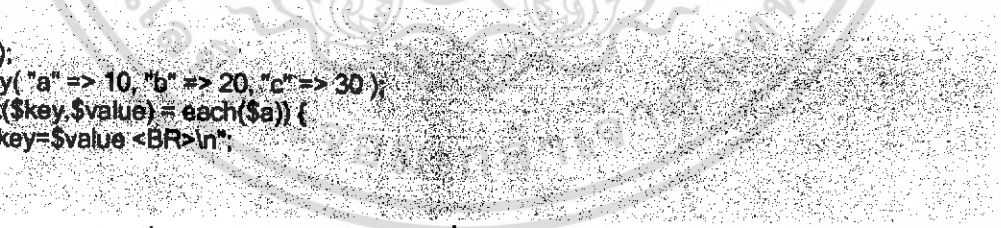
เราอาจจะสร้างอาร์เรย์เป็นสองมิติก็ได้ เช่น

ตัวอย่างที่ 2.51

```
<?
$countries = array (
  "thailand"  => array ( "zone" => "Asia", "D_NAME" => ".th"),
  "malasia"  => array ( "zone" => "Asia", "D_NAME" => ".my"),
  "india"      => array ( "zone" => "Asia", "D_NAME" => ".in"),
  "holland"  => array ( "zone" => "Europe", "D_NAME" => ".nl"),
  "france"    => array ( "zone" => "Europe", "D_NAME" => ".fr")
);
echo "domain name=".$countries["thailand"]["D_NAME"]."<BR>\n";
?>
```

การใช้คำสั่ง each และ list สำหรับ associative array

ถ้าเราต้องการจะเข้าถึงข้อมูลแต่ละคู่ที่ถูกเก็บอยู่ใน associative array เราอาจจะใช้วิธีเรียกผ่านฟังก์ชัน each() และ list() ตามตัวอย่างที่ 2.52 ต่อไปนี้



```
<?
unset($a);
$a = array( "a" => 10, "b" => 20, "c" => 30 );
while (list($key,$value) = each($a)) {
  echo "$key=$value <BR>\n";
}
?>
```

ฟังก์ชัน each() จะอ่านข้อมูลทีละคู่จากอาร์เรย์แบบเชื่อมโยงมาแล้วส่งไปยังฟังก์ชัน list() ซึ่งจะทำหน้าที่แยกเก็บ ซึ่งในกรณีนี้คือ เก็บไว้ในตัวแปร $key และ $value หลังจากนั้น เราก็สามารถนำค่าของตัวแปร ไปใช้งานตามที่ต้องการได้

การนิยามและสร้างฟังก์ชันโดยผู้ใช้ (User-defined functions)

ถ้าเราต้องการสร้างฟังก์ชันขึ้นมาใช้งานเองก็ทำได้ โดยเฉพาะอย่างยิ่งในกรณีที่เราต้องการจะ ใช้ชุดคำสั่งเหล่านั้นบ่อยครั้ง เราก็จัดเก็บเป็นฟังก์ชัน เพื่อให้เรียกใช้ได้สะดวก และยังช่วยให้การเขียนโปรแกรมง่ายขึ้นด้วย

การสร้างฟังก์ชันขึ้นใช้เองทำได้โดยใช้โครงสร้าง

```
function function_name ($arg1, $arg2, .., $argN) {
 ....
}
```

และฟังก์ชันจะให้ค่ากลับคืนหรือไม่ก็ได้ ถ้าต้องการให้ค่ากลับคืนจากการทำงานของฟังก์ชัน ก็จะใช้คำสั่ง return นอกจากนั้น PHP ยังสนับสนุน default parameter ด้วย

ตัวอย่างที่ 2.53 เช่น การหาค่าสัมบูรณ์ของตัวเลข

```
<?
function myabs ($x) {
  if ($x < 0)
    return -$x;
}
echo myabs(-6),"<BR>\n";
echo myabs(-4+2.034),"<BR>\n";
?>
```

การหาค่าดังกล่าวของตัวเลขใดๆ เราสามารถใช้ฟังก์ชัน abs() หรือเราเขียนขึ้นเองก็ได้ตามตัวอย่างข้างบน

การสร้างฟังก์ชันแบบเรียกตัวเอง (recursive function)

ตัวอย่างที่ 2.54 การหาค่าแฟลทอเรียล n!

```
<?
function factorial ($n) {
  if ( ($n == 0) || ($n == 1) )
    return 1;
  else
    return $n*factorial($n-1);
}
echo factorial(4);
?>
```

เงื่อนไขก็ใช้ฟังก์ชัน factorial() จากตัวอย่างข้างบน คือ $n จะต้องเป็นตัวแปรที่เก็บค่าที่เป็นเลขจำนวนเต็ม และไม่เป็นลบ ถ้าเราต้องการจะเขียนฟังก์ชันให้มีความปลอดภัยในการใช้งาน เราก็อาจจะเพิ่มเงื่อนไข เพื่อตรวจเช็คดูก่อนว่า ผู้ใช้ผ่านค่าของตัวแปรที่ตรงตามต้องการหรือไม่ เช่น ไม่ผ่านค่าที่เป็นสตริงค์ หรือเป็นเลขทศนิยม หรือค่าที่เป็นลบ เป็นต้น

ตัวอย่างที่ 2.55 การค้นหาข้อมูลแบบ Binary Search ในอาร์เรย์ที่มีการเรียงข้อมูลจากน้อยไปมาก

```
<?
function binSearch(&$key,&$array, $left, $right)
{
 $mid = ceil( ($left + $right) / 2 );
 if ($left > $right)
   return -1;
 if ($array[$mid] == $key)
   return $mid;
 else if ($key < $array[$mid])
   return binSearch($key, $array, $left, $mid-1); // recursive call
 else
   return binSearch($key, $array, $mid+1, $right); // recursive call
}
$num=100;
$key = randint(0, $num);
for($i=0; $i < $num; $i++) {
 $sorted_array[$i] = $i+1;
}
echo binSearch(13, $sorted_array, 0, $num);
?>
```

ตัวอย่างที่ 2.56 การสร้างสตริงค์แบบสุ่มอีกแบบหนึ่งซึ่งอาจจะนำไปใช้ในการสร้าง one-time password (OTP)

```
<?
function randomToken($len) {
 srand( date("s") );
 $chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
 $chars.= "1234567890!@#$%^&*()";
 $ret_str = "";
 $num = strlen($chars);
 for($i=0; $i < $len; $i++) {
  $ret_str.= $chars[rand()%$num];
 }
 return $ret_str;
}
echo randomToken(13)," ";
?>
```

หมายเหตุ: การกำหนดค่า seed สำหรับฟังก์ชัน srand() นอกจะใช้ date("s") เป็นตัวกำหนดค่าแล้ว เราอาจจะใช้ฟังก์ชันอื่นก็ได้ เช่น srand((double)microtime()*1000000);

การใช้ตัวแปรแบบ global ภายในฟังก์ชัน

บางครั้งเราไม่ต้องการที่จะผ่านตัวแปรเป็นอาร์กิวเมนต์ของฟังก์ชัน เพื่อนำไปใช้ภายในฟังก์ชันเหล่านั้น ก็จะทำได้โดยการแจ้งใช้ตัวแปรที่มีชื่อเหมือนตัวแปรภายนอกที่เราต้องการใช้ ให้เป็น global หรือใช้ผ่านตัวแปรที่เป็นอาร์เรย์ของ PHP ที่มีชื่อว่า $GLOBALS ดังตัวอย่างที่ 2.57 ต่อไปนี้

```
<?
$a = 10;
```

```
$b = 20;
function getMin ( ) {
 global $a, $b;
 if ($a < $b)
  return $a;
 else
  return $b;
}
function getMin2 () {
 if ($GLOBALS["a"] < $GLOBAL["b"])
  return $GLOBALS["a"];
 else
  return $GLOBALS["b"];
}
echo getMin()."<BR>\n";
echo getMin2()."<BR>\n";
?>
```

ในกรณีนี้เราต้องการจะใช้ตัวแปร $a และ $b ซึ่งอยู่นอกฟังก์ชัน getMin() เพื่อเช็คดูว่า ค่าของตัวแปรใดมีค่าน้อยกว่ากัน ถ้าเราไม่แจ้งใช้ global $a, $b; ตามตัวอย่างแล้ว $a และ $b จะกลายเป็นตัวแปรภายในแม้ว่าจะชื่อเหมือนกันตัวแปรภายนอกที่มีอยู่แล้วก็ตาม ทำให้ได้ผลการทำงานไม่ถูกต้องตามที่ต้องการ ฟังก์ชัน getMin() อีกรูปแบบหนึ่ง โดยไม่ใช้ตัวแปรแบบ global ภายในฟังก์ชัน และใช้วิธีผ่านค่าแทน ตัวอย่างที่ 2.58

```
<?
$a = 10;
$b = 20;
function getMin ($a, $b) {
 if ($a < $b)
    return $a;
 else
    return $b;
}
echo getMin($a, $b)."<BR>\n";
?>
```

การตัวแปรแบบ static ภายในฟังก์ชัน

สมมุติว่า เราต้องการจะใช้ตัวแปรภายในฟังก์ชัน และสามารถเก็บค่าไว้ได้ตลอดเวลา โดยไม่สูญหายไปทุกครั้งที่มีการเรียกใช้ฟังก์ชัน ในกรณีนี้เราจะแจ้งใช้ตัวแปรให้เป็นแบบ static ตามตัวอย่างที่ 2.59 ต่อไปนี้

```
function MyFunc() {
  static $num_func_calls = 0;
  echo "my function\n";
  return ++$num_func_calls;
}
```

ทุกครั้งที่มีการเรียกใช้ฟังก์ชันดังกล่าว ตัวแปรชื่อ $num_func_calls ซึ่งมีค่าเริ่มต้นเป็นศูนย์ในตอนแรก จะเพิ่มค่าที่เก็บขึ้นทีละหนึ่ง

การผ่านค่ากลับคืนมากกว่าหนึ่งจากฟังก์ชัน

โดยปรกติแล้วเราไม่สามารถผ่านค่ากลับคืนจากฟังก์ชันได้มากกว่าหนึ่ง แต่อย่างไรก็ตาม ยังมีวิธีการหนึ่งที่ช่วยแก้ปัญหาดังกล่าวได้ วิธีนี้คือ เก็บค่าต่างๆที่ต้องการจะใช้เป็นค่ากลับคืนไว้ใน array แล้วใช้ array นั้นเป็นค่ากลับคืน และผู้เรียกใช้ฟังก์ชันสามารถใช้ฟังก์ชัน list() อ่านค่าเหล่านั้นได้

การอ่านตัวแปรจากภายนอกที่ได้จากการ Web browser โดยวิธี GET หรือ POST

สมมุติว่า เรามีฟอร์มสำหรับให้ผู้ใช้ป้อนชื่อ (login) และรหัสผ่าน (password) จากนั้นก็ส่งมายัง Webserver และใช้สคริปต์ PHP เป็นตัวจัดการกับข้อมูลที่ส่งมาโดยวิธีการแบบ POST ตามตัวอย่างที่ 2.60

ภายในสคริปต์ login.php เราสามารถอ่านข้อมูลที่ส่งมาได้ ในการนี้ ที่เราสนใจคือ ค่าของ login และ password ที่อยู่ในฟอร์ม และสามารถจะอ่านข้อมูลเหล่านั้นได้ เพราะ PHP จะเก็บข้อมูลไว้ในตัวแปรชื่อ $login และ $password ตามลำดับ

การตรวจดู webbrowser ของผู้มาเยือนว่าเป็นตัวไหน

อีกตัวอย่างหนึ่งที่แสดงให้เห็นการใช้ตัวแปรแบบ global ซึ่งเป็นตัวแปรที่ตัวแปลชุดคำสั่ง PHP ได้สร้างขึ้น ทุกครั้งที่ทำงาน หนึ่งในตัวแปรนั้นคือ $HTTP_USER_AGENT

การสร้างและใช้งานคลาส (class) และออปเจค (object)

ภาษาแบบ scripting language ในปัจจุบันหลายๆภาษาก็สนับสนุนการเขียนโปรแกรมเชิงวัตถุด้วย ตัวอย่างเช่น Perl และ PHP ก็รวมอยู่ในนั้นด้วย แม้ว่าจะไม่ซับซ้อนเหมือนอย่างภาษาซีพลัสพลัสหรือจาวาก็ตาม

คลาสคือโครงสร้างที่ประกอบด้วยสมาชิก (class members) หรือคุณสมบัติ (properties) ตามแต่จะเรียก และ ฟังก์ชันสมาชิก (member functions) การนิยามคลาสขึ้นมาใช้งานจะเริ่มด้วย class { ... } โดยข้างในจะมีส่วนของตัวแปรสมาชิก และฟังก์ชันสมาชิกตามลำดับ ฟังก์ชันที่มีชื่อเดียวกับคลาสจะเรียกว่า class constructor ทุกครั้งที่มีการสร้างออปเจคจากคลาสโดยใช้คำสั่ง new ฟังก์ชันที่ทำหน้าที่เป็น class constructor ก็จะถูกเรียกมาทำงานก่อนทุกครั้ง ประโยชน์ของการใช้งานก็เช่น ใช้กำหนดค่าเริ่มต้น หรือเตรียมพร้อมก่อนที่จะเริ่มใช้ออปเจค

ลองดูตัวอย่าง การเขียนคลาสสำหรับแบบข้อมูลเชิงนามธรรม (Abstract Data Type) ที่เรียกว่า stack การทำงานของ stack ก็เป็นดังนี้ ถ้าเราใส่ข้อมูลเข้าไป ข้อมูลเหล่านั้นก็จะถูกเก็บไว้เสมือนกับว่า วางซ้อนกันจากข้างล่างขึ้นข้างบน ถ้าเราจะดึงข้อมูลออกมาใช้ก็จะได้ข้อมูลที่อยู่ข้างบนสุด ซึ่งก็คือข้อมูลที่เราใส่เข้าไปครั้งล่าสุดนั่นเอง หน้าที่ของ stack ที่สำคัญก็มีเช่น

ตารางที่ 2.16 หน้าที่ของ stack ที่สำคัญ

push()	ใส่ข้อมูลไว้ใน stack
pop()	ดึงข้อมูลออกมา
is_empty()	ตรวจดูว่า stack มีข้อมูลอยู่อีกหรือไม่
get_size()	หาจำนวนของข้อมูลที่ถูกเก็บไว้ใน stack

ตารางที่ 2.16 หน้าที่ของ stack ที่สำคัญ

การใช้คำสั่ง include และ require

คำสั่งทั้งสองเอาไว้แทรกเนื้อหาจากไฟล์อื่นที่ต้องการ ข้อแตกต่างระหว่าง include และ require อยู่ตรงที่ว่า ในกรณีของการแทรกไฟล์ใช้ชื่อต่างๆ กันมากกว่าหนึ่งครั้งโดยใช้ลูป คำสั่ง require จะอ่านเพียงแค่ครั้งเดียว คือไฟล์แรก และจะแทรกไฟล์นี้เท่านั้นไปตามจำนวนครั้งที่วนลูป ในขณะที่ include สามารถอ่านได้ไฟล์ต่างๆ กันตามจำนวนครั้งที่ต้องการ

Regular Expression

Regular Expression หรือเรียกย่อๆว่า Regex หมายถึง รูปแบบของลำดับ หรือกลุ่มของสัญลักษณ์ ที่ใช้แทนลำดับ หรือกลุ่มของอักขระตามที่ต้องการ

เราใช้สัญลักษณ์ [ ] (square brackets) เพื่อกำหนดขอบเขตของกลุ่มตัวอักขระหลายตัวที่ใช้เป็นตัวเลือก เช่น สมมุติว่า เราต้องการจะเขียนรูปแบบที่ใช้แทนตัวอักขระหนึ่งตัว อะไรก็ได้จาก {a,e,i,o,u} เราก็จะเขียนว่า [aeiou] โดยจะเรียงลำดับก่อนหลังอย่างไรก็ได้ เช่น [eioua] ให้ผลเหมือนกับ [aeoui] หรือ ถ้าเราต้องการเขียน รูปแบบเพื่อใช้แทนตัวขระหนึ่งตัวที่เป็นตัวเลขตัวใดตัวหนึ่งจาก 0 ถึง 9 เราก็เขียนว่า [0123456789] หรือจะเขียนแบบสั้นๆใหม่ได้เป็น [0-9] หรืออีกตัวอย่างหนึ่ง ถ้าเราต้องการจะเขียนนิพจน์แบบ regex ขึ้นมาเพื่อใช้แทนอักขระตัวใดตัวหนึ่งที่เป็นได้ทั้งตัวพิมพ์ใหญ่หรือเล็กในภาษาอังกฤษหรือตัวเลขระหว่าง 0 ถึง 9 เราก็เขียนว่า [A-Za-z0-9]

ตารางที่ 2.17 Regular Expression

[aeiou]	ตัวอักขระตัวหนึ่งจาก {a,e,i,o,u} ตัวไหนก็ได้
[0-9]	ตัวอักขระตัวหนึ่งจาก {0,1,...,9} ตัวไหนก็ได้
[A-Za-z0-9]	ตัวอักขระตัวหนึ่งจาก {A,B,..,Z, a, b, ... , z, 0, 1, ... 9} ตัวไหนก็ได้

ตารางที่ 2.17 Regular Expression

บทที่ 3

การออกแบบและพัฒนา

แนวทางการออกแบบ

3.1ส่วนของการเลือกภาษาที่ใช้ในการพัฒนาโปรแกรมในโครงงานนี้ เป็นการออกแบบพัฒนาโปรแกรมออกแบบติดตั้งโปรแกรมหัวล่อฟ้า บนเว็บแอบพลิเคชั่น ซึ่งในส่วนของการออกแบบนั้นได้มีส่วนที่มีการกำหนดค่าไว้อยู่แล้ว และนำมาใช้งานเป็น Server ที่ใช้งานสำหรับรันโปรแกรมออกแบบติดตั้งหัวล่อฟ้านั้นมี ค่ากำหนดของการทำงานของ Server ดังนี้

SITE INFORMATION

Configuration	
Site Name	suntec.co.th
Email Contact	tassanai_nai@yahoo.com
Administrator User Name	
Services & Options	Domain Aliasing
	✓Bandwidth Monitor (for cycle beginning Thu Feb 01 2007) Used / Threshold: 124 MB / 18000 MB (130682441 bytes / 18874368000 bytes) Cycle Start: Day 1 of month
	✓Analog Web/FTP Log Analyzer
	Webalizer Log Analyzer
	✓Apache Web Server: http://www.suntec.co.th/ Domain Preview: http://61.47.2.8/suntec.co.th/
	✓CGI: details Script Alias: cgi-bin
	Tomcat 4
	Generate Web Logs
	✓Server Side Includes

Modperl for Apache
FrontPage Server Extensions
✓Subdomains
Miva Merchant
✓File Manager
✓Backup/Restore
✓POP3 + Imap Server
✓Email: mail.suntec.co.th
~~Vacation Auto-Responder~~
Mailing Lists (Majordomo)
✓Spam Filtering
✓SquirrelMail Web-based Email: Read Email
✓MailScanner / Virus Scanner
✓Scan incoming mail
Scan outgoing mail
Development Tools
OpenSSH Secure Shell
Telnet
✓FTP: ftp.suntec.co.th
Anonymous FTP
✓MySQL Database Administrator: suntec Number of Databases: 3 Database Prefix: suntec_co_th_

ตารางที่ 3.1 แสดง SITE INFORMATION

จึงทำให้ต้องเลือกใช้ภาษา PHP ในการเขียนโปรแกรมออกแบบติดตั้งหัวล่อฟ้า แต่เนื่องจากการทำงานของ PHP ยังไม่ยืดหยุ่นเพียงพอกับความต้องการของ Requirement ได้ จึ้งใช้หลักการทำงานของ AJAX มาช่วยในการทำงาน ซึ่งในส่วนนี้จึงต้องเรียนรู้รูปแบบการทำงานของ AJAX เช่น

- หลักการทำงานของ XMLHTTPRequest
- ภาษา JavaScript ที่นำมาใช้เป็นตัวกลางดึงข้อมูลจาก XMLHTTPRequest

3.2 ส่วนของการออกแบบการวางหัวล่อฟ้า

ในส่วนนี้ของโครงงาน ได้ศึกษาเกี่ยวกับการทำงานของหัวล่อฟ้าวิธีการติดตั้งหัวล่อฟ้าและเก็บผลลัพท์ของรูปแบบตึกในโปรเจคต่างๆที่ได้เคยทำการติดตั้งหัวล่อฟ้านำมาเป็นข้อมูลในการสร้างแบบตึก โดยสามารถสรุปรูปแบบตึกและอาคารที่พบเจอซึ่งมีความสำคัญมากสุดอยู่ 4 แบบคือ

แบบที่ 1

ตึกรูปสี่เหลี่ยม

รูปที่ 3.1 ตึกรูปสี่เหลี่ยม

แบบที่ 2

ตึกรูปตัว L

รูปที่ 3.2 ตึกรูปตัว L

แบบที่ 3

ตึกรูปสี่เหลี่ยมมีหลังคาเป็นสามเหลี่ยมหน้าจั่ว

รูปที่ 3.3 ตึกรูปสี่เหลี่ยมมีหลังคาเป็นสามเหลี่ยมหน้าจั่ว

แบบที่ 4

ตึกรูปสี่เหลี่ยมมีหลังคาเป็นสามเหลี่ยมมุมฉาก

รูปที่ 3.4 ตึกรูปสี่เหลี่ยมมีหลังคาเป็นสามเหลี่ยมมุมฉาก

3.3 ส่วนของการออกแบบฐานข้อมูลของรัศมีหัวล่อฟ้า

ในส่วนนี้ได้ใช้สูตรการคำนวณหารัศมีและขนาดความสูงของเสาเข้ามาใช้ในการสร้างฐานข้อมูล

โดยมีสูตรดังนี้

สูตรการคำนวณหาค่ารัศมีป้องกันฟ้าผ่าของหัวล่อฟ้า Stormaster ESE terminal

จากห้องทดลอง ซึ่งได้มาตราฐานจากประเทศฝรั่งเศส.

$Rp = \sqrt{h(2D-h)+\Delta T(2D+\Delta T)}$ เมื่อความสูงเสามากกว่า 5 เมตร

• เมื่อ ΔT คือค่าการส่งผ่านกระแสไฟฟ้าในอากาศ

Stormaster-ESE-50 = จะมีค่า ΔT (μs)50

Stormaster-ESE-60 = จะมีค่า ΔT (μs) 60

• h = คือค่าความสูงของเสาหัวล่อฟ้าที่บริเวณที่ต้องการได้รับการป้องกันจากอันตรายจากฟ้าผ่า(มีค่าเป็นเมตร)

• D = คือค่าตัวแปลคงที่ ซึ่งจะมีค่าขึ้นอยู่กับ Level ที่เลือกมาคำนวณ

ซึ่ง ค่าค่านี้ได้รับมาตรฐานของ standard NF C 17-102.

D = 20m for protection level 1 (High Protection)

D = 45m for protection level 2 (Medium protection)

D = 60m for protection level 3 (Standard protection)

PROTECTION RADIUS (M) - (Rp)

ซึ่งออกมาเป็นตารางเพื่อใช้เป็นฐานข้อมูลในการคำนวณตำแหน่งการวางหัวล่อฟ้าดังนี้

PROTECTION RADIUS (M) - (Rp)									
h = height of Stormaster terminal above area to be protected (m)	2	4	5	6	10	15	20	45	60
Protection Level 1 (High Protection)									
Stormaster 50	28	55	68	69	69	70	70	-	-
Stormaster 60	32	64	79	79	79	80	80	-	-
Protection Level 2 (Medium Protection)									
Stormaster 50	35	69	86	87	88	90	92	95	-
Stormaster 60	40	78	97	97	99	101	102	105	
Protection Level 3 (Standard Protection)									
Stormaster 50	38	76	95	96	98	100	102	110	110
Stormaster 60	44	87	107	107	109	111	113	120	120

รูปที่ 3.5 แสดงตารางข้อมูลของหัวล่อฟ้า

username	password
saksit	46015374
admin	admin

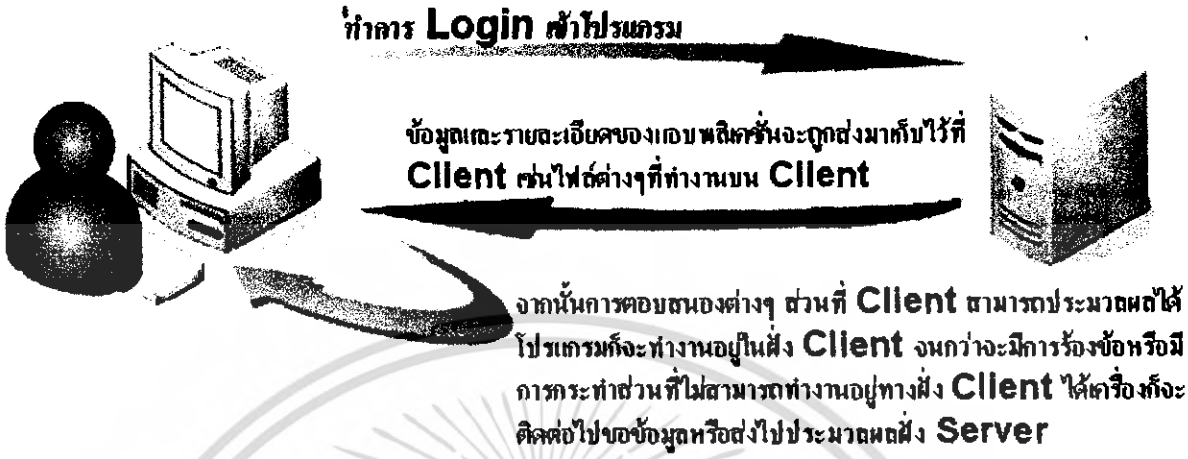
รูปที่ 3.6 แสดงตารางข้อมูลของ User

model	height	radius
Stormaster 50 (standard)	2	38
Stormaster 50 (standard)	4	76
Stormaster 50 (standard)	5	95
Stormaster 50 (standard)	6	96
Stormaster 50 (standard)	10	98
Stormaster 50 (standard)	15	100
Stormaster 50 (standard)	20	102
Stormaster 50 (standard)	45	110
Stormaster 50 (standard)	60	110
Stormaster 60 (standard)	2	44
Stormaster 60 (standard)	4	87
Stormaster 60 (standard)	5	107
Stormaster 60 (standard)	6	107
Stormaster 60 (standard)	10	109
Stormaster 60 (standard)	15	111
Stormaster 60 (standard)	20	113
Stormaster 60 (standard)	45	120
Stormaster 60 (standard)	60	120
Stormaster 50 (medium)	2	35
Stormaster 50 (medium)	4	69
Stormaster 50 (medium)	5	86
Stormaster 50 (medium)	6	87
Stormaster 50 (medium)	10	88
Stormaster 50 (medium)	15	90
Stormaster 50 (medium)	20	92
Stormaster 50 (medium)	45	95
Stormaster 60 (medium)	2	40
Stormaster 60 (medium)	4	78
Stormaster 60 (medium)	5	97
Stormaster 60 (medium)	6	97

รูปที่ 3.7 แสดงตารางข้อมูลขนาดและรัศมีหัวล่อฟ้า

3.4 หลักการทำงานของโปแกรม และUSE CASE SEQUENCE DIAGRAM

หลักการทำงานของโปรแกรมในส่วนต่างๆ



รูปที่ 3.8 แสดงหลักการทำงานระหว่าง Client กับ Server

อธิบาย File ที่สำคัญของโปรแกรม

ไฟล์ที่ทำงานบนฝั่ง Client

1. Index.php คือไฟล์ที่ทำหน้าที่เป็นส่วนแสดงและตอบสนองส่วนต่างๆของโปรแกรม Tool bar ต่างๆ รับค่า Input ขนาดแผนที่ กำหนดมุมมองของทิศทางตำแหน่งตึก

2.Building1.js, Building2.js, Building3.js, Building4.js คือไฟล์ที่ใช้สำหรับรับค่าที่ผู้ใช้งานกรอกมาเพื่อคำนวณสร้างรูปตัวตึกแสดงสู่หน้าจอ และเก็บค่าต่างๆของตัวตึก เช่นค่า X,Y บนแผนที่ ค่า W1,W2,W3,H1,H2 ซึ่งทำโดยการเก็บขนาดและทิศทางตึกส่งไปให้ wz_jsgraphics.js คำนวณเพื่อวาดภาพตึกออกมาเป็นแบบ Iso metrix และ แบบ Topiew ทั้ง 4 มุม

3. Wz_jsgraphics.js คือไฟล์คำนวณการสร้างภาพตัวตึก

4. Dragdrop.js คือส่วนที่เก็บฟังก์ชั่น ที่ทำหน้าที่แปลงพิกัดหน้าจอเป็นพิกัดแผนที่แบบ และพิกัดแผนที่แบบเป็นพิกัดหน้าจอ รวมกระทั่งฟังก์ชั่นลาก Object ต่างๆ ฟังก์ชั่น Redraw และฟังชั่น ฯลฯ

5. XML.js คือไฟล์ที่เป็นฟังก์ชั่น ที่นำข้อมูลรายละเอียดของแบบ แปลงเป็น XML String และปลง XML String เป็นแผนที่ตึกและตำแหน่งทิศทางของตึกบนแบบ

6. Pillar.js คือไฟล์ที่มีฟังก์ชั่นที่วาดขนาดเสารัศมีหัวล่อฟ้าและกำหนดรายละเอียดต่างๆของหัวล่อฟ้า

ไฟล์ที่ทำงานบนฝั่ง Server

1.Select.php คือไฟล์ที่เชื่อมต่อกับฐานข้อมูลหัวล่อฟ้าและเป็นไฟล์ที่จัดการที่ส่ง หน้าเว็บเพจมายัง client ให้ผู้ใช้ได้เลือก รุ่นและขนาดของหัวล่อฟ้า

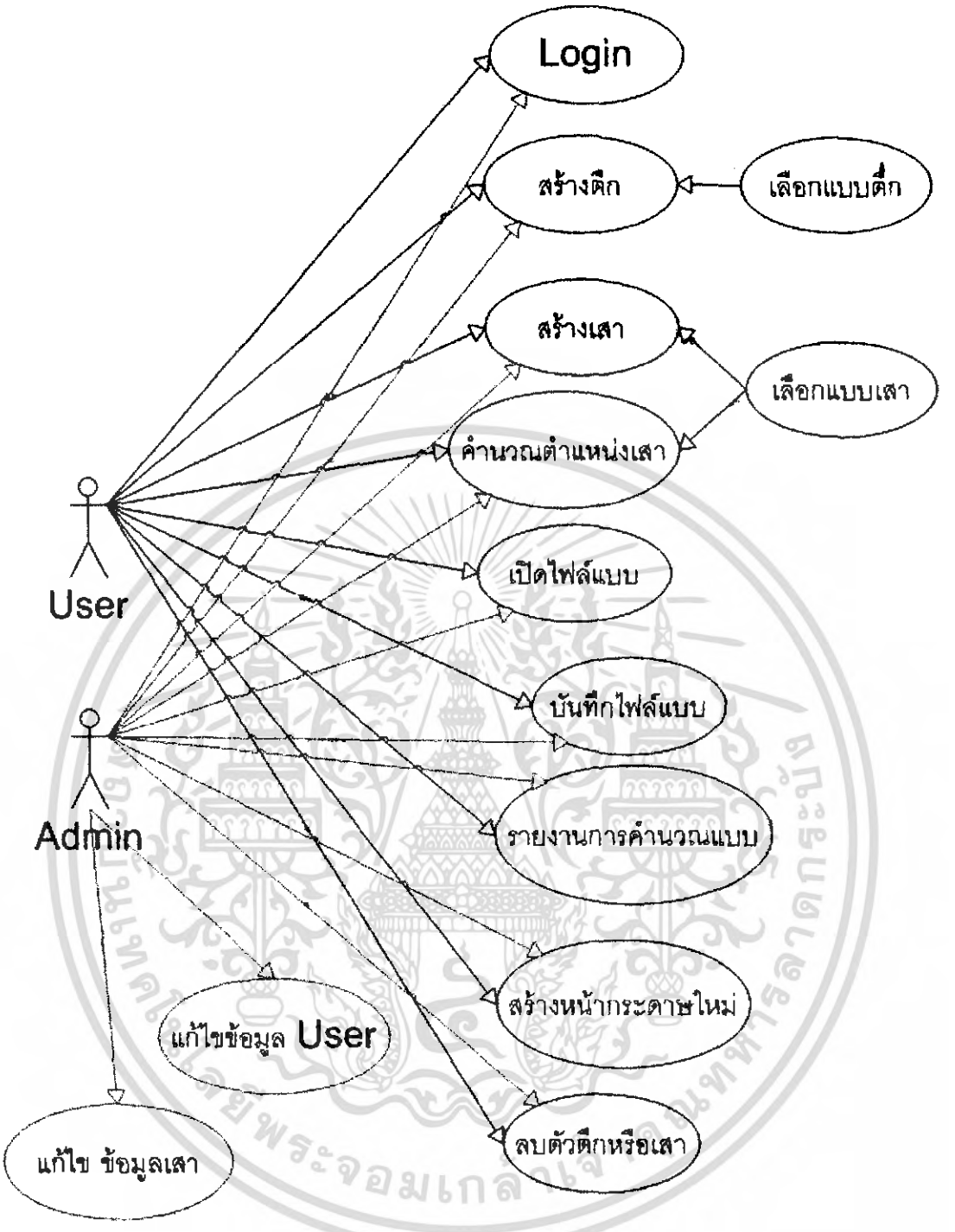
2.Cal.php คือไฟล์ที่มีฟังก์ชั่นการคำนวณวางหัวล่อฟ้าบนตึกที่สร้าง

3.Report.php คือไฟล์ที่ทำหน้าที่รับค่าแบบตึกที่คำนวณมาแล้วมาจัดเรียงเพื่อแสดงผลให้ผู้ใช้งานได้ดู และสามารถพิมพ์ออกมาได้ โดยจะส่งข้อมูลต่างๆที่ได้รับมาจาก XML.js ไปยัง Getpic.php เพื่อได้รูปภาพมาจัดเรียงและแสดงผล

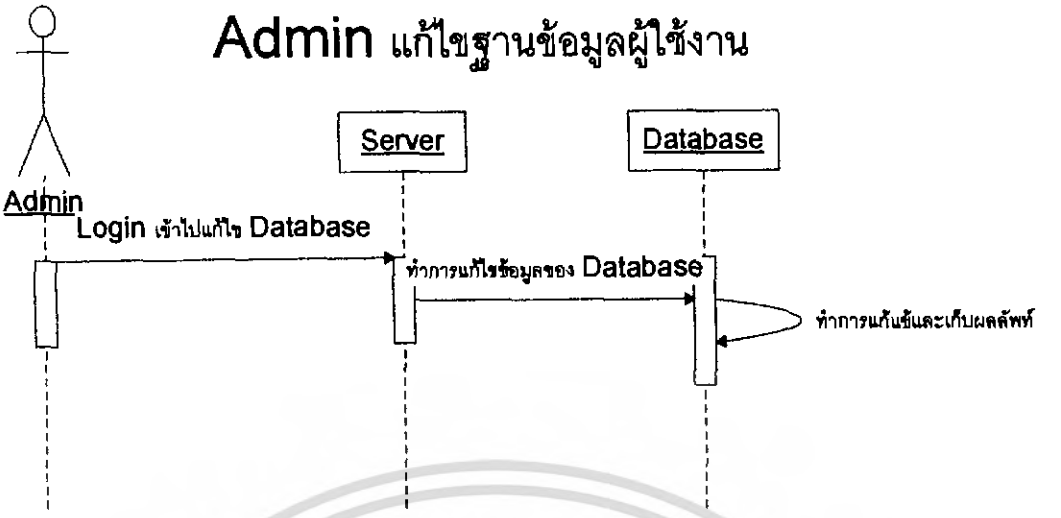
4. Getpic.php คือไฟล์ที่ทำหน้าที่นำข้อมูลมาสร้างเป็นรูปภาพ

5. Sendfile คือไฟล์ที่ทำหน้าที่ส่งไฟล์ที่ผู้ใช้ต้องการบันทึก ไปให้ฝั่ง Client เพื่อให้ผู้ใช้บันทึกข้อมูลลงเครื่องคอมพิวเตอร์ของตน

6. Getfile.php คือไฟล์ที่ทำหน้าที่เรียกไฟล์ที่ผู้ใช้งานต้องการเปิดมาใช้งาน



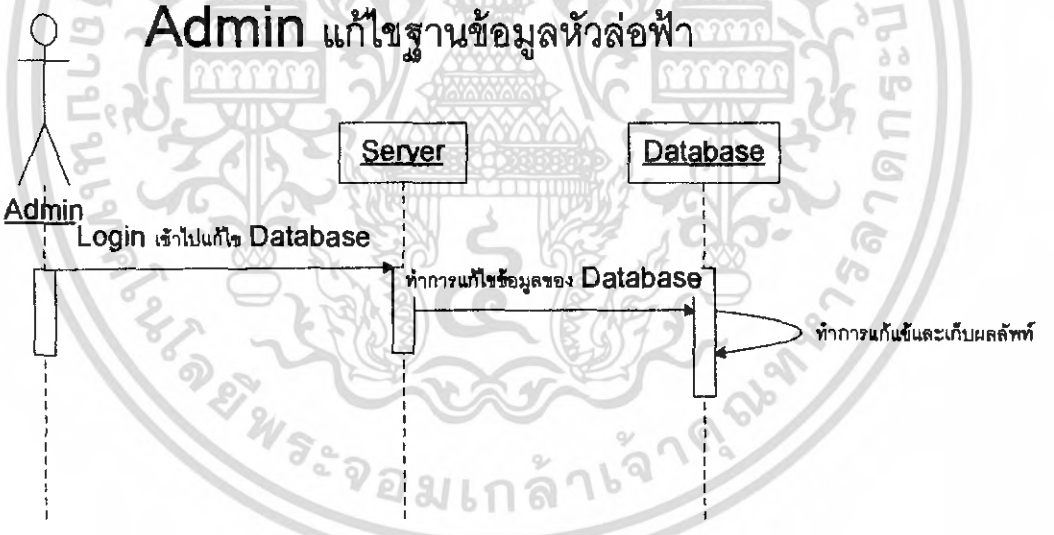
รูปที่ 3.9 แสดง USE CASE การทำงานของโปรแกรม



รูปที่ 3.10 แสดง SEQUENCE DIAGRAM การแก้ไขฐานข้อมูลผู้ใช้งาน

แก้ไขข้อมูล User

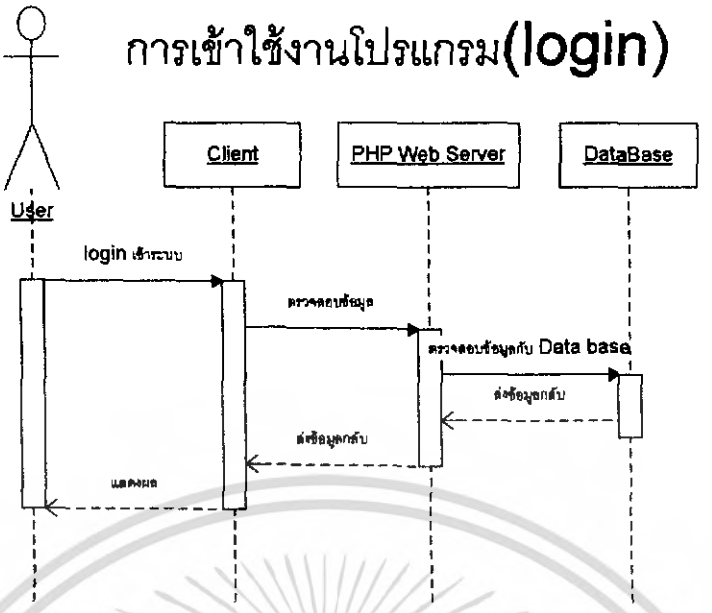
1. *Admin login* สู่เว็บเซิร์ฟเวอร์เพื่อเชื่อมต่อฐานข้อมูลของ User
2. Admin ทำการแก้ไข User



รูปที่ 3.11 แสดง SEQUENCE DIAGRAM แก้ไขข้อมูลหัวล่อฟ้า

แก้ไขข้อมูลหัวล่อฟ้า

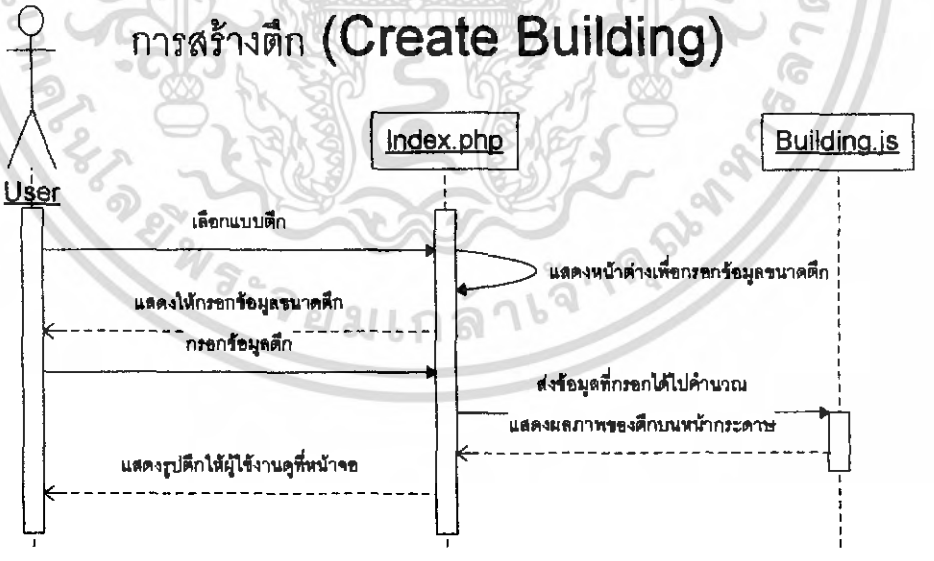
1. *Admin login* สู่เว็บเซิร์ฟเวอร์เพื่อเชื่อมต่อฐานข้อมูลของหัวล่อฟ้า
2. Admin ทำการแก้ไขขนาดเสาและรัศมีหัวล่อฟ้า



รูปที่ 3.12 แสดง SEQUENCE DIAGRAM การเข้าใช้งานโปรแกรม

LOGIN

1. User ใส่ Username,Password ที่ฝั่ง ไคล์เอนต์
2. ไคล์เอนต์ติดต่อกับ เซิฟเวอร์ร้องขอการ Login ของ User
3. เซิร์ฟเวอร์ทำการตรวจสอบสิทธิ์ของ User โดยเทียบจากฐานข้อมูล
4. เซิร์ฟเวอร์ ส่งผลกลับมายัง ไคลเอนต์



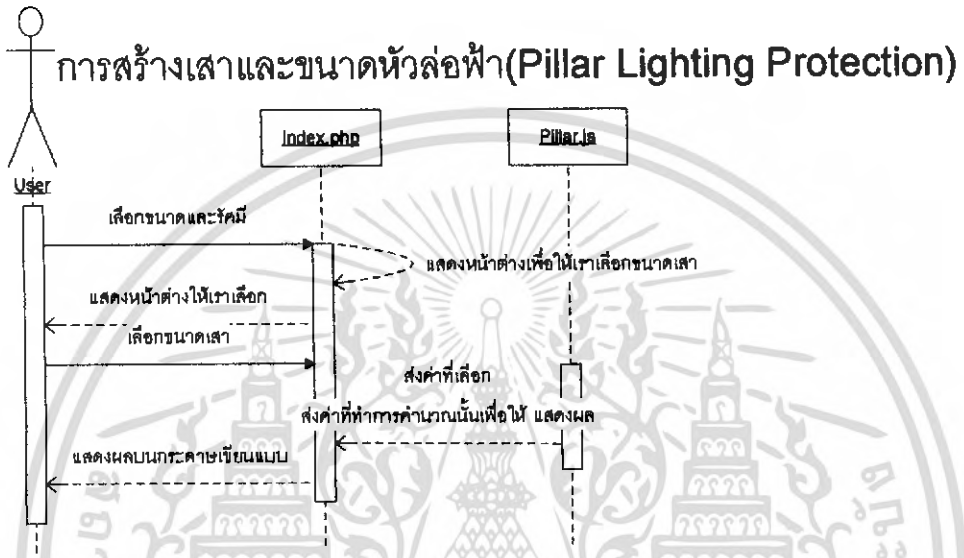
รูปที่ 3.13 แสดง SEQUENCE DIAGRAM การสร้างตึก

สร้างตึก

1. User เลือกแบบตึกในหน้า Index.php
2. User กรอกรายละเอียดขนาดของตึกที่ต้องการสร้าง

3. จากนั้น index.php จะส่งข้อมูลไปที่ Building.js คำนวณและสร้างตัวตึกเพื่อส่งมาแสดงที่หน้า Index.php

4.หน้า index.php แสดงภาพให้ผู้ใช้เห็นบนแบบ



รูปที่ 3.14 แสดง SEQUENCE DIAGRAM การสร้างเสาหัวล่อฟ้า

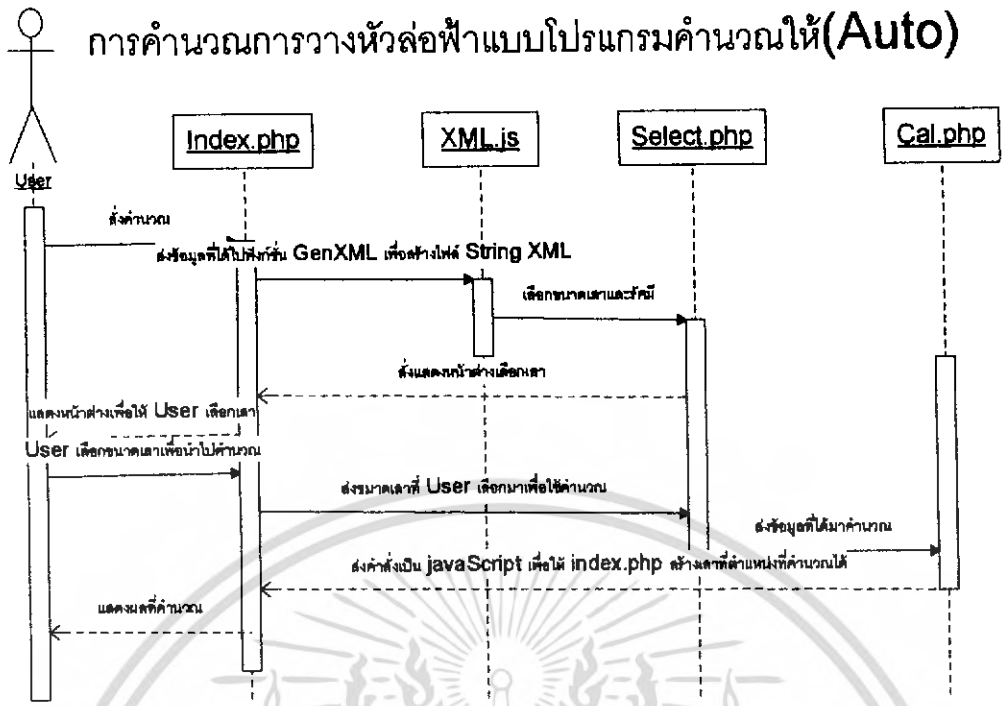
สร้างเสา

1.User เลือกขนาดเสาจากหน้า Index.php

2. User สั่งสร้างเสา ข้อมูลที่เลือกจะถูกส่งไปที่ไฟล์ Pillar.js เพื่อวาดขนาดและรัศมีของเสาส่งกลับมาให้ dragdrop.js

3. User เลือกวางตำแหน่งเสา

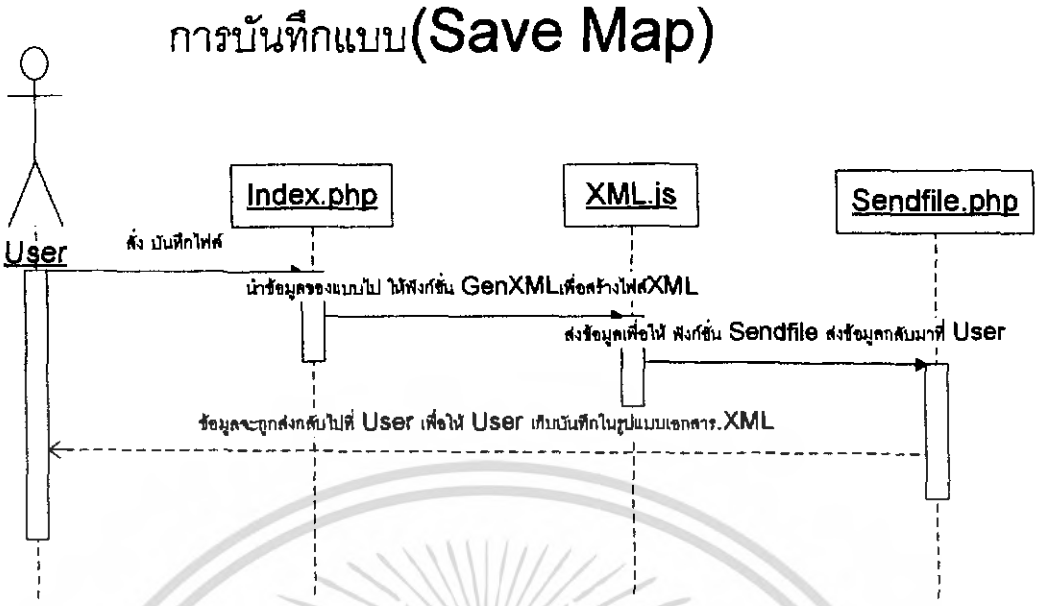
4. Dragdrop.js และ Pillar.js สั่ง ให้ index.php แสดงเสาตามตำแหน่งที่ระบุ



รูปที่ 3.15 แสดง SEQUENCE DIAGRAM การคำนวณตำแหน่งเสา

คำนวณตำแหน่งเสา

1. User เลือกคำสั่ง Calculate เพื่อการทำนวณตำแหน่งเสา ข้อมูลของแบบจะถูกส่งไปที่ XML.js เพื่อนำข้อมูลมาสร้างเป็น XML string และจากนั้นจะถูกส่งไปเก็บไว้ที่ Select.php
2. Select.php ส่งหน้าเพจให้ User เลือกระบุรุ่นเสา
3. User เลือกรุ่นเสาจากนั้นกดยืนยันการคำนวณ ข้อมูลทั้งหมดที่ถูกเลือกและแบบเสาจะถูกส่งจาก Select.php ไปยัง Cal.php เพื่อคำนวณหาตำแหน่งวางเสาบนดึก
4. Cal.php ส่งคำสั่งเป็น JavavScript เพื่อให้ Index.php แสดงผล
5. Index.php แสดงผลลัพท์ที่ได้มาให้ User



*เนื่องจาก Javascript สามารถสร้างไฟเพื่อส่งกลับให้ User บันทึกได้จึงส่งไฟล์ไปให้ Server เพื่อ Server จะสร้างไฟล์และส่งกลับมาบันทึกที่ Client แทน

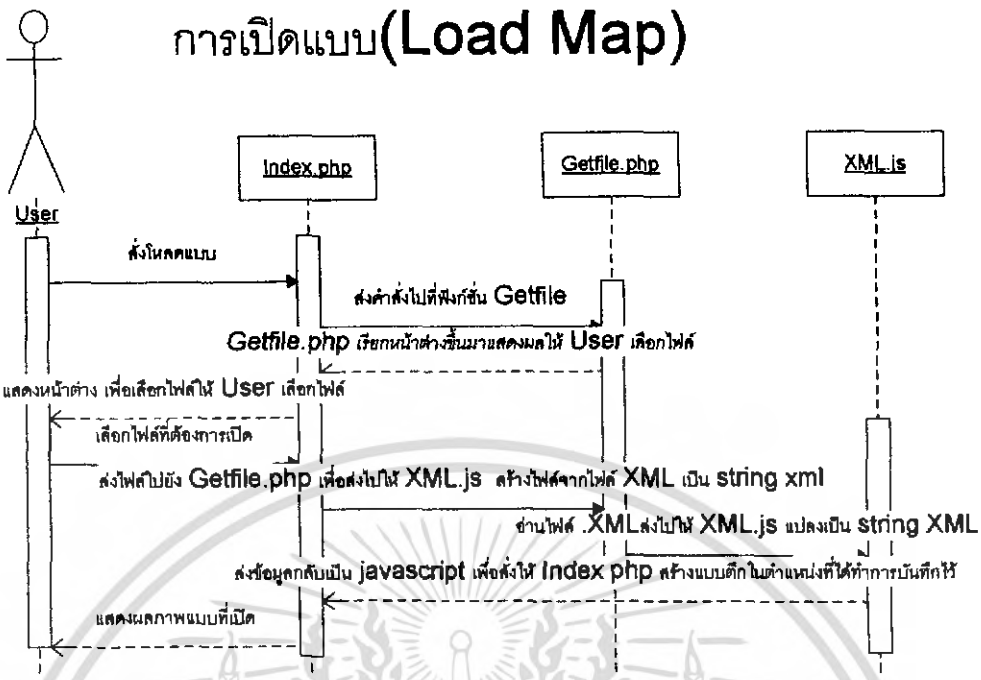
รูปที่ 3.16 แสดง SEQUENCE DIAGRAM การบันทึกไฟล์แบบ

บันทึกไฟล์แบบ

1. User สั่งบันทึกแบบ ในหน้า Idex.php จากนั้นข้อมูลแบบดึกจะถูกส่งไปให้ XML.js สร้างไฟล์ให้อยู่ในรูปแบบภาษา XML จากนั้นจะส่งไปให้ Sendfile.php เพื่อสร้างไฟล์ส่งกลับไปบันทึกที่ฝั่งไคลเอนต์

2. Sendfile.php ส่งข้อมูลไปฝั่ง ไคลเอนต์เพื่อให้ User ระบุที่เก็บและยืนยันการบันทึก

3. User ยืนยันการบันทึก



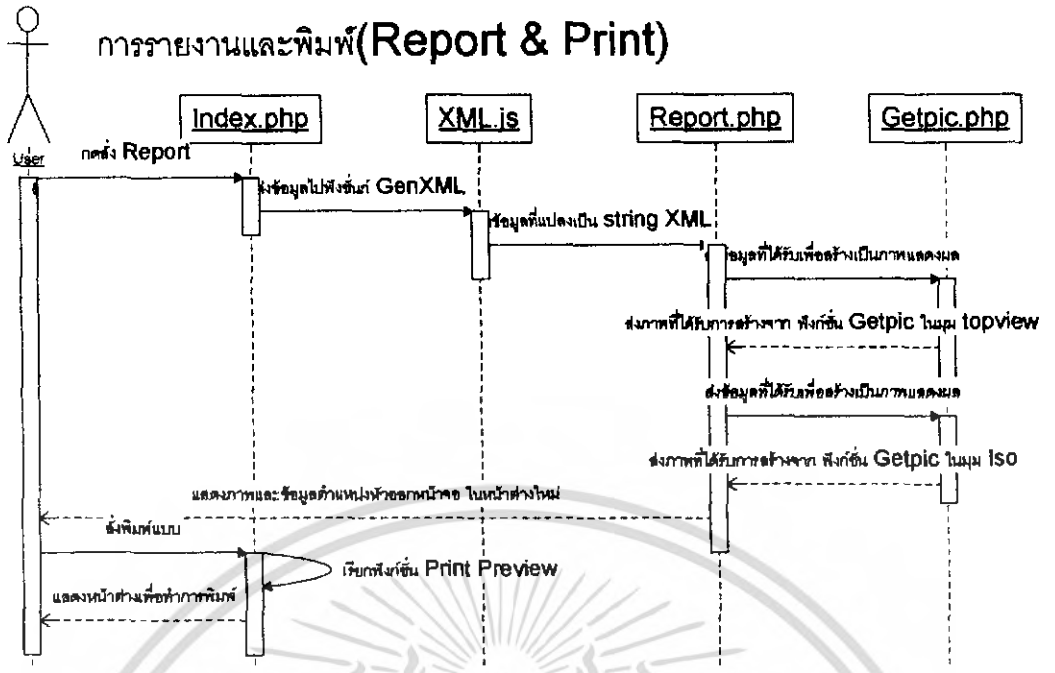
รูปที่ 3.17 แสดง SEQUENCE DIAGRAM การเปิดไฟล์แบบ

เปิดไฟล์แบบ

1. User สั่งเปิดไฟล์แบบ ที่หน้า Index.php จากนั้น Getfile.php จะส่งหน้าต่างให้ User ระบุที่เก็บไฟล์ ที่จะเปิด

2. User เลือกไฟล์ที่จะเปิดแล้วยืนยัน ข้อมูลถูกส่งไปที่ Getfile.php จากนั้นจะถึงส่งไปยัง XML.js เพื่อ แปลงโค้ดที่ได้มาเป็น คำสั่งโดยส่งไปให้ Index.php สร้างภาพแบบที่ได้ทำการเปิดมาบนหน้าจอ

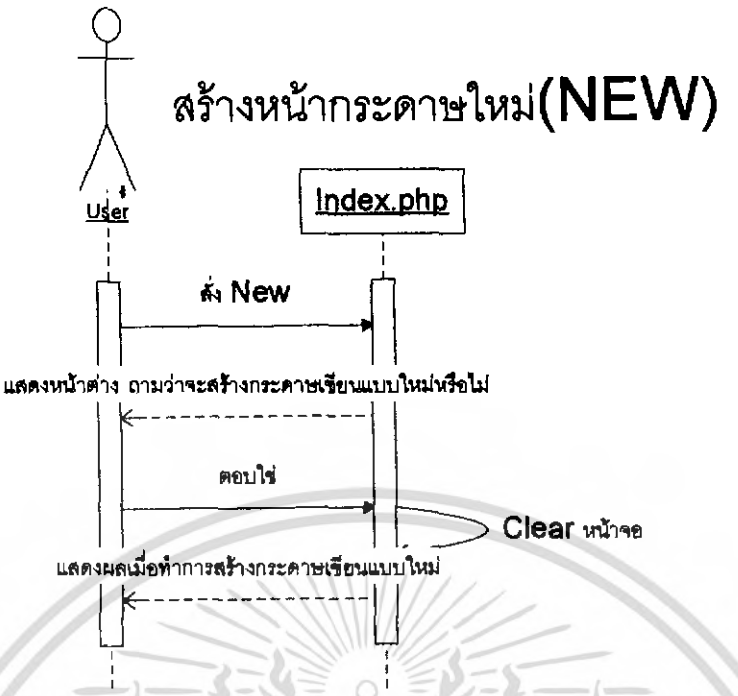
3.Index.php แสดงผลแบบให้ User



รูปที่ 3.18 แสดง SEQUENCE DIAGRAM รายงานการคำนวณแบบ

รายงานการคำนวณแบบ

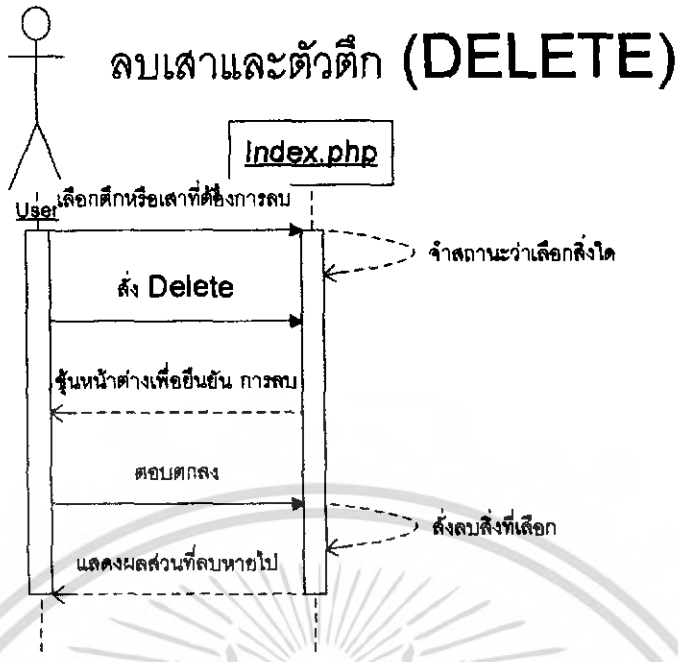
1. User สั่ง รายงานผล Index.php ส่งข้อมูลแบบไป XML.js เพื่อแปลง ได้ดเป็น XML string
2. จากนั้นข้อมูลทั้งหมดจะถูกส่งไปให้ Report.php คำนวณ
3. Report.php ส่งข้อมูลไปสร้างภาพแสดงผลที่ Getpic.php จากนั้น Getpic.php จะสร้างภาพขึ้นและส่งกลับมาที่ Report.php เพื่อจัดรูปแบบหน้าที่จะรายงาน
4. Report.php แสดงหน้าต่างให้ User ดูรายละเอียด
5. User สั่ง Print



รูปที่ 3.19 แสดง SEQUENCE DIAGRAM การสร้างหน้ากระดาษใหม่

สร้างหน้ากระดาษใหม่

1. User สั่งสร้างกระดาษใหม่
2. User ยืนยันการสร้างกระดาษใหม่
3. Index.php ทำการสร้างกระดาษใหม่



รูปที่ 3.20 แสดง SEQUENCE DIAGRAM การลบตัวตึกหรือเสา

ลบตัวตึกหรือเสา

1. User สั่งลบตัวตึกหรือเสาที่ต้องการ
2. User ยืนยันการลบ
3. Index.php ทำลบตัวตึกหรือเสาที่ User ต้องการออก

บทที่ 4

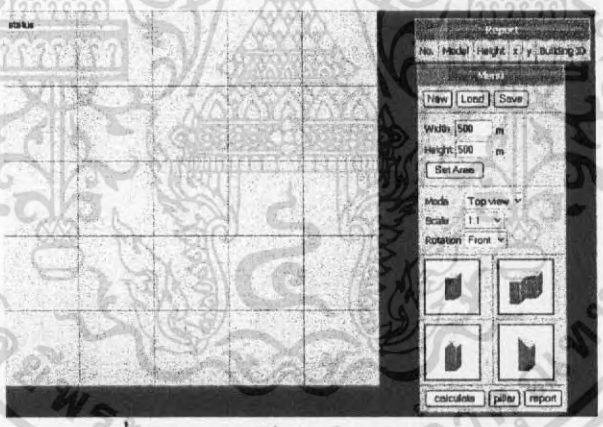
การทดลองและผลการทดลอง

การทดลองโปรแกรม

การทดลองโปรแกรม โดยทำการอัพโหลดโปรแกรมขึ้น Server จริงแล้วทำการทดลองออกแบบระบบติดตั้งหัวล่อฟ้า

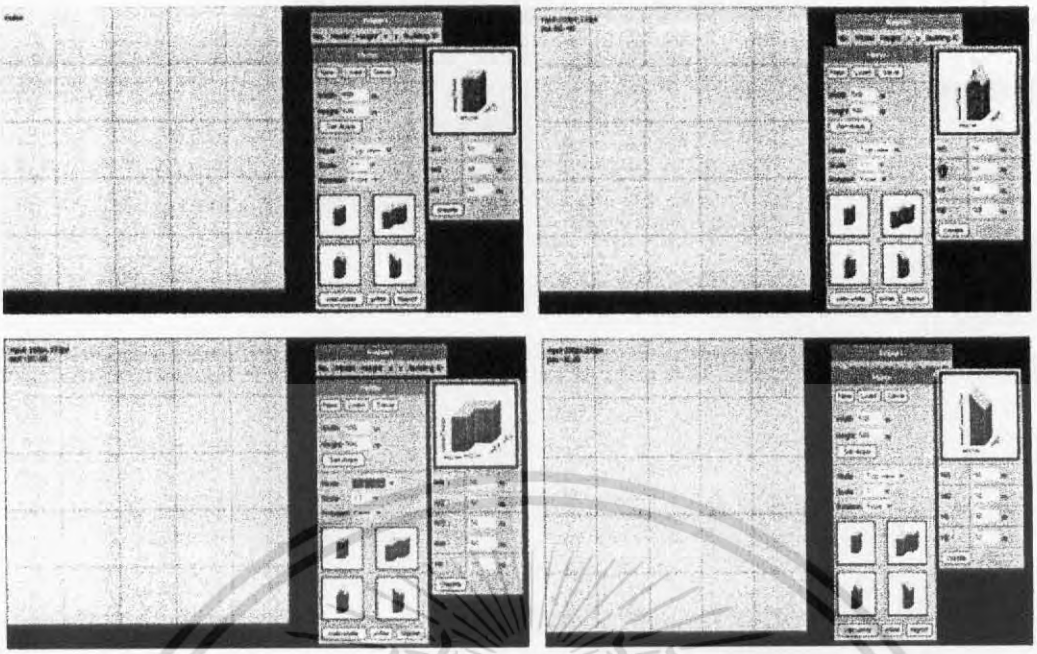
รูปที่ 4.1แสดงหน้า Login

ทดลอง login เข้าสู่โปรแกรม โปรแกรมออกแบบระบบติดตั้งหัวล่อฟ้า



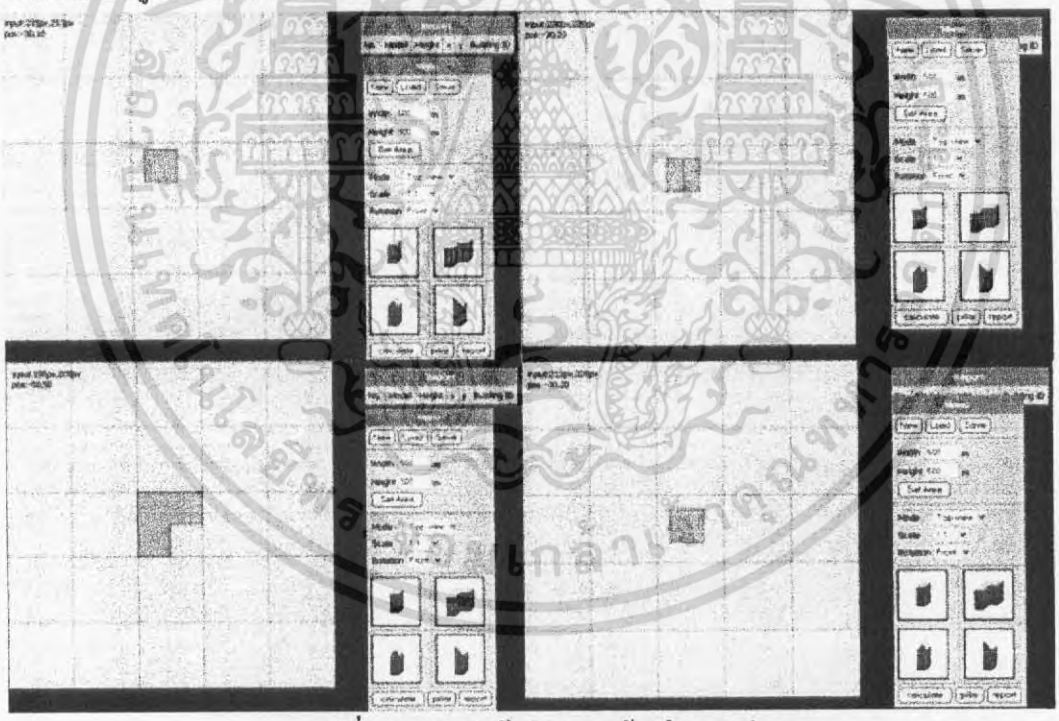
รูปที่ 4.2 แสดงหน้าต่างกระดาษเขียนแบบ

หน้าตา index.php จะแสดงรายละเอียดต่างๆ โดยทำการ โหลดข้อมูลจากฝั่ง server มาเก็บไว้ที่เครื่อง Client



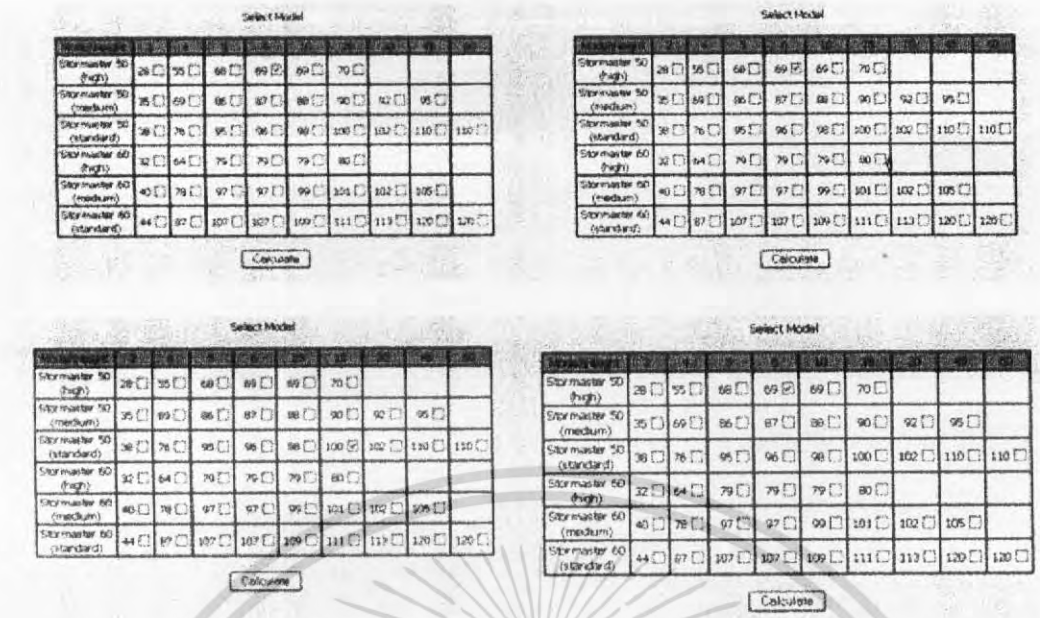
รูปที่ 4.3 แสดงหน้าต่างการกรอกขนาดตึกในแบบต่างๆ

ทดลองเลือกรูปแบบตึกเพื่อกรอกขนาดของตัวตึก



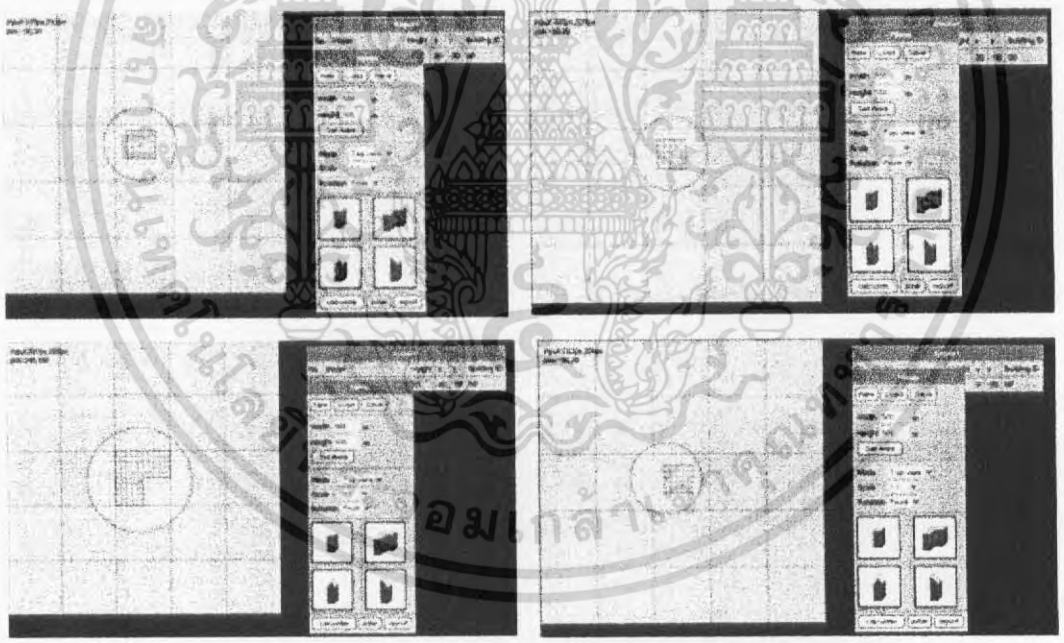
รูปที่ 4.4 แสดงหน้าต่างการสร้างตึกแบบต่างๆ

เมื่อกรอกข้อมูลขนาดตึกเสร็จแล้วจะได้ตัวตึกแล้วนำมาวางบนพื้นที่ที่ต้องการ



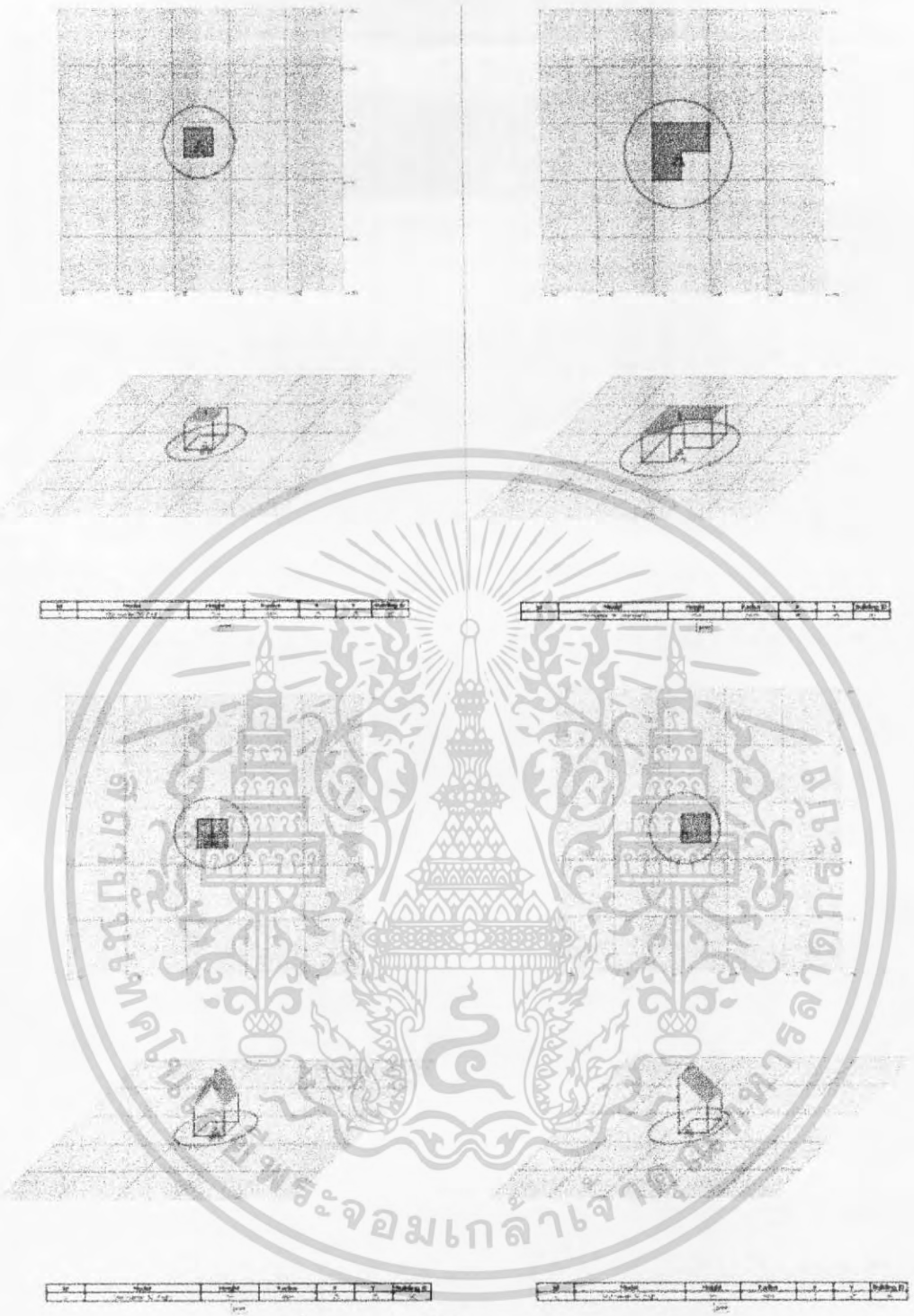
รูปที่ 4.5 แสดงหน้าต่างเลือกขนาดและรัศมีของหัวล่อฟ้า

ทดลองเลือกขนาดเสาเพื่อใช้ในการคำนวณ



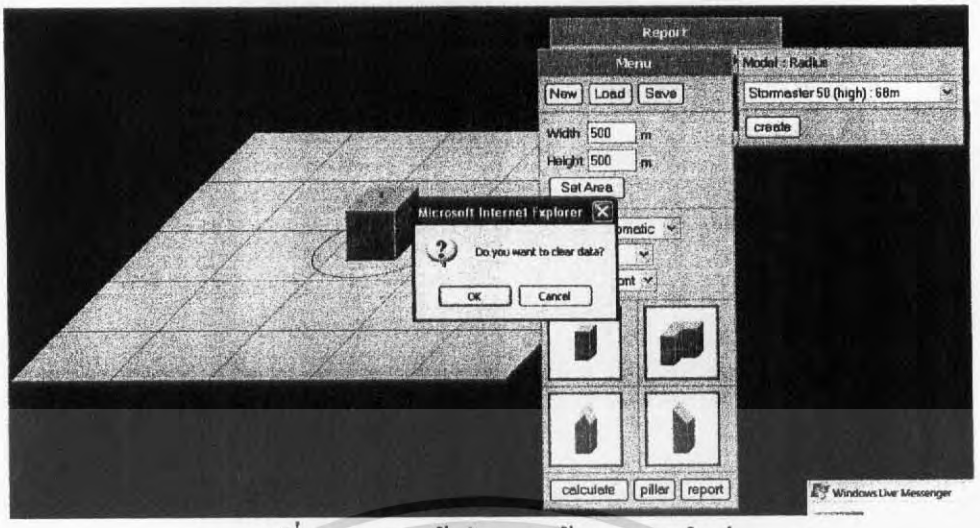
รูปที่ 4.6 แสดงหน้าต่างการคำนวณหาตำแหน่งหัวล่อฟ้าในแบบต่างๆ

แสดงผลการคำนวณในส่วนต่างๆ



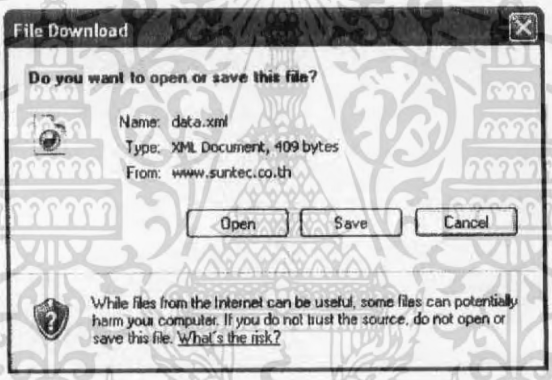
รูปที่ 4.7 แสดงหน้าต่างรายงานผลในแบบต่างๆ

ทดลองรายงานผลการออกแบบ ในมุมมอง top view และ Iso metric และแสดงรายละเอียดตำแหน่งของเสาบนตัวอาคาร



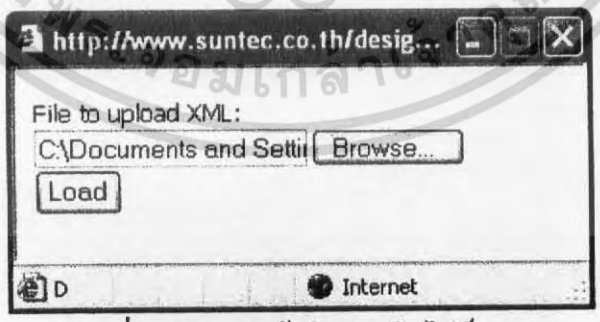
รูปที่ 4.8 แสดงหน้าต่างการสร้างกระดาษใหม่

ทดลองสร้างหน้ากระดาษใหม่



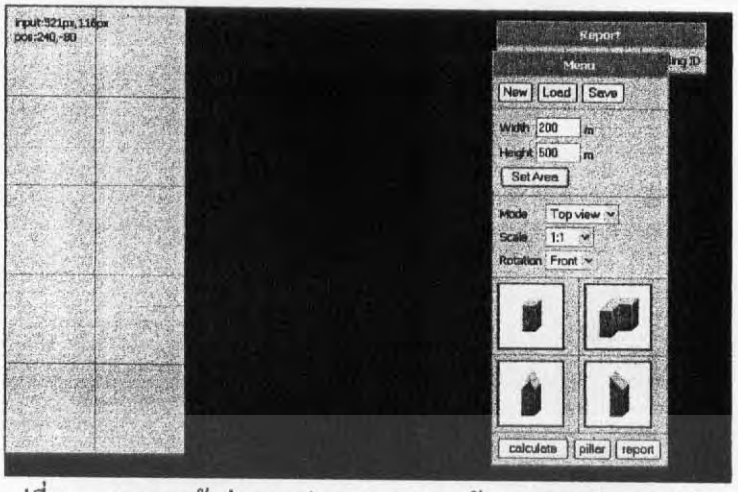
รูปที่ 4.9 แสดงหน้าหน้าต่างการบันทึกข้อมูล

ทดลองทำการบันทึกไฟล์แบบที่เขียน



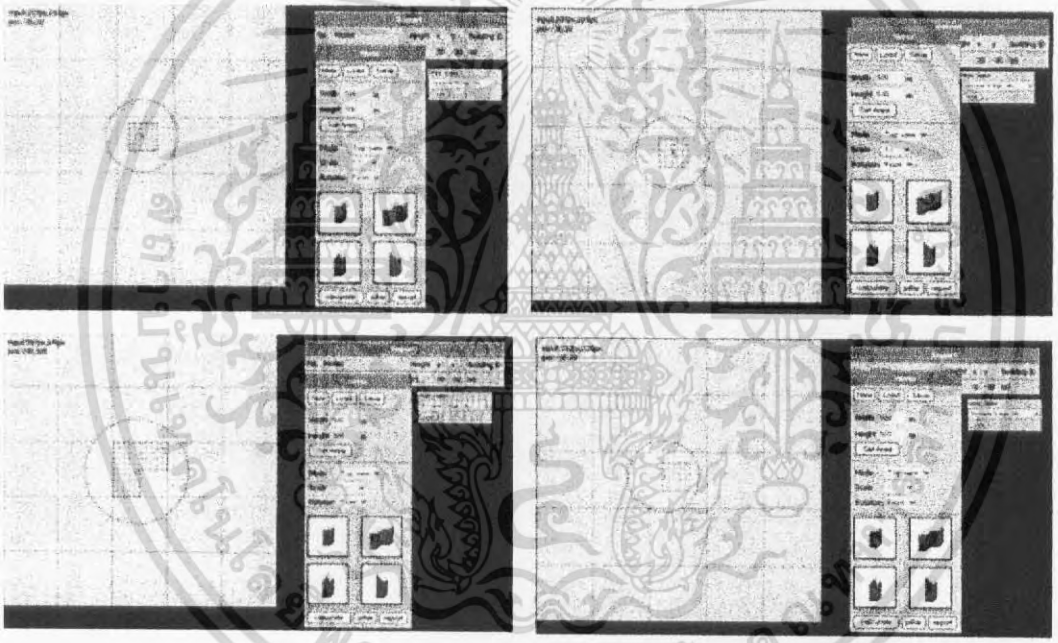
รูปที่ 4.10 แสดงหน้าต่างการเปิดไฟล์แบบ

ทดลองโหลดไฟล์ที่ได้บันทึกไว้ในเครื่อง



รูปที่ 4.11 แสดงหน้าต่างการกำหนดขนาดหน้ากระดาษเขียนแบบใหม่

ทดลองกำหนดขนาดหน้ากระดาษเขียนแบบใหม่



รูปที่ 4.12 แสดงหน้าต่างการเลือกตำแหน่งวางหัวล่อฟ้าเอง

ทดลองกำหนดตำแหน่งและขนาดเสาเอง

บทที่ 5

สรุปผลการดำเนินโครงงาน

ในหัวข้อนี้จะกล่าวถึง บทสรุป, ปัญหาที่พบในระหว่างการดำเนินโครงงาน, แนวทางการแก้ไข และ แนวทางในการพัฒนาในอนาคต

5.1 บทสรุป

จากการศึกษาออกแบบและทดลองในโครงงานนี้ จะทำการออกแบบตัวอาคารที่จะทำการติดตั้งหัวล่อฟ้าและคำนวณตำแหน่งการวางหัวล่อฟ้า โดยผ่านเว็บแอปพลิเคชั่นซึ่งช่วยให้พนักงานบุคลากรที่เกี่ยวข้องได้รับความสะดวกสบายมากยิ่งขึ้น และยังส่งผลให้ระบบรักษาความปลอดภัยในประเทศไทยพัฒนามากขึ้นตามด้วย

ซึ่งโครงงานนี้เป็นโครงงานที่นำเอาเทคโนโลยีการรักษาความปลอดภัยมาสร้างเป็นแอปพลิเคชั่นเพื่อ ออกแบบการติดตั้งหัวล่อฟ้า เพื่อพัฒนาระบบและเทคโนโลยีของ การรักษาความปลอดภัย ซึ่งในส่วนการออกแบบแอปพลิเคชั่นนั้นได้ใช้ภาษา PHP ในการเขียนแอปพลิเคชั่นโดยใช้ภาษา PHP ทำงานในส่วนต่างๆที่โปรแกรมทำงานที่ฝั่ง Server และใช้ภาษา JavaScript และ XML มาทำงานในส่วนของโปรแกรมที่ทำงานฝั่ง Client และเนื่องจากการทำงานของ ภาษา PHP นั้นมีความยืดหยุ่นในการแสดงผลน้อย จึงได้ใช้เทคโนโลยีหลักการทำงานของ AJAX เข้ามาช่วย เพื่อให้ได้โปรแกรมที่มีประสิทธิภาพมากขึ้น แต่ในส่วนของโปรเจคนี้ก็ยังมีหลายๆจุดที่สามารถพัฒนาเพิ่มเติมได้ในอนาคต แต่เนื่องจากประสบการณ์และความรู้ของผู้พัฒนายังไม่เพียงพอ แต่ก็สามารถพัฒนาโปรแกรมให้สามารถใช้ได้ในมาตรฐานที่ บริษัทผู้ใช้แอปพลิเคชั่นนี้สามารถใช้งานจริงได้ และสิ่งที่ได้ประโยชน์นั้น คือทำให้ได้เรียนรู้ระบบรักษาความปลอดภัยของตัวอาคารสถานที่ และเข้าใจในหลักการทำงานของ AJAX มากขึ้น และสามารถนำความรู้ที่ได้จากการพัฒนาโครงงานนี้ไปใช้งานได้ในอนาคต

5.2 ปัญหาที่พบในระหว่างการดำเนินโครงงาน

1. ข้อมูลที่ใช้ในการศึกษาเช่นในส่วน รูปแบบการทำงานของหัวล่อฟ้า และวิธีการออกแบบการติดตั้ง นั้นค่อนข้างมีข้อมูลและตัวอย่างให้ศึกษาน้อยมาก
2. ผู้จัดทำยังเป็นมือใหม่ในการทำงานด้านนี้ ทำให้การทำงานเป็นไปได้ไม่สะดวกมากนักเพราะยังต้องศึกษาในส่วนต่างๆเพิ่มเติมอีกมากทำให้งานนี้เดินไปทีละก้าวอย่างช้าๆ
3. ขอบเขตการพัฒนาแอปพลิเคชั่นแคบ มีปัญหาทำให้ใช้ภาษาในการเขียนแอปพลิเคชั่นได้น้อย
4. การทำงานของ Web Browser กับ Server มีข้อจำกัดในการประมวลผล เนื่องจากถ้าไม่มีการส่ง Response กลับใน 30 วินาทีจะทำให้ Web Browser แจ้งว่าเป็น Error

5. เนื่องจาก ภาษา JavaScript ไม่มีฟังก์ชันการสร้างภาคสามมิติ จึงทำให้ต้องสร้างฟังก์ชั่นเพื่อคำนวณค่าและวาดภาพออกมาเป็นภาพสามมิติ ทำให้การประมวลผลช้า

5.4 แนวทางในการพัฒนาในอนาคต

1. พัฒนาการใช้งานให้ง่ายและสามารถยืดหยุ่นต่อแบบอาคารที่มีแบบแตกต่างกันได้มากขึ้น
2. พัฒนาให้การแสดงผลมีความสวยงามและเหมือนจริง
3. พัฒนาการประมวลผลการทำงานให้ประมวลผลได้เร็วขึ้น
4. ปรับปรุง ให้โปรแกรมสามารถนำไฟล์ที่อยู่ในรูปแบบของโปรแกรมเขียนแบบอื่นๆให้สามารถนำเข้ามาออกแบบติดตั้งหัวล่อฟ้าได้
5. ให้มีฟังก์ชันในส่วนต่างมากขึ้น เช่น สามารถคำนวณราคาเพื่อยื่นใบเสนอราคาได้

บรรณานุกรม

[1] Ing Peter Hasse ,Can lightning protection be improved by ESE,Technical University *Illmenau/Germany in* ETZ,*Heft* 3-4 /2002.

[2] สมาคมวิศวกรรมสถานแห่งประเทศไทยในพระบรมราชูปถัมภ์,มาตรฐานการป้องกันฟ้าผ่าสำหรับสิ่งปลูกสร้าง,พิมพ์ครั้งที่ 1 ตุลาคม 2543,หน้า 1-16.

[3] วิชา ศิริธรรมจักร์ 2549 **Web Programming ด้วย AJAX และ PHP** สำนักพิมพ์ เคทีพี กรุงเทพฯ

[4] พวงร้อย คำเรียง, ฟ้าผ่า, ออนไลน์, http://www.vcharkarn.com/include/article/showarticle.php?Aid=78&page=3, วันที่เข้าถึง 10 มกราคม 2548.

[5] Sandy Kieft, Lightning rod types, ออนไลน์, http://www.ee.nmt.edu/~langmuir/l-rods.html, วันที่เข้าถึง 10 มกราคม 2548, 20 มกราคม 2548,โครงการฟิสิกส์และวิศวกรรม ฝ่ายวิจัยและพัฒนาเทคนิค องค์การโทรศัพท์แห่งประเทศไทย

[6] **เรียนรู้การเขียนเว็บเพจด้วยภาษา PHP ด้วยตนเอง ภายใน 5.5 สัปดาห์**

http://www.bcoms.net/php/php38.asp

http://php.deeserver.net/

http://www.hotscripts.com/PHP/

http://www.thaicreate.com/index.php?modules=dir/explorer.php